Trustonic

MobiCore®

MobiCore Validation Test Suite

Version 1.2.0.0 Build manually

Contents

Chapter 1

MobiCore Validation Test Suite

Disclaimer

The validation suite does not perform any system or performance or stability test. The focus of the validation suite is to verify the MobiCore functionality.

Introduction

This documentation contains the test specification of the MobiCore Validation Test Suite. The MobiCore Validation Test Suite can be used by OEMs to perform testing with MobiCore binaries. The document describes all the test cases that are implemented in the Trustlet Connector (TLC) 'ValidationTestSuite'.

These tests are implemented in one binary: -> *ValidationTestSuite*

A Trustlet is needed to interact with the Validation Test Suite.

-> Its binary file is called *05010000000000000000000000000000.tlbin*

Also a Driver is needed to interact with the Validation Test Suite.

-> Its binary file is called *02040000000000000000000000000000.tlbin*

To execute the tests please follow the instruction in chapter 'MobiCore Validation Test Suite' of the file 'Readme.pdf'.

Glossary

A

ADB - Android Debug Bridge

Activation - Before a Service Provider can install and run Trustlets, MobiCore needs to be activated on the device by the System Owner. The use case Root Registration is equivalent to Activation. Activation changes the life cycle state of MobiCore to a fully

functional state.Note: before activation MobiCore may be available to support certain device security features, but will not be able to support Service Provider's trusted services.

API - Application Programming Interface

ARM - Advanced RISC Machines

B

BIT - Binary Digit

C

Container - Containers are hierarchical data structures holding content management relevant data and are associated to the various roles in the MobiCore ecosystem. MobiCore containers are the Root Container, Service Provider Containers and Trustlet Containers.

CM - Content Management

CM Trustlet (CMTL) - MobiCore System Trustlet and is responsible for content management operations in cooperation with the NWd Provisioning Agent.

CPU - Central Processing Unit

D

DCI - Driver Control Interface

DDK - Device Driver Kit, SW development kit for creating device drivers for specific operating system.

DS-5 - ARM Development Studio 5

F

FC - FastCall from the SWd to the MobiCore

FCI - FastCall interface of the MobiCore

FIQ - Fast Interrupt Request, concept of the ARM CPU for a high priority interrupt that can even interrupt the IRS of an IRQ. In a TZ system FIQs are used as secure interrupts handeld in the SWd while IRQs are handled in the NWd.

H

HAL - Hardware Abstraction Layer

I

IEC - International Electrotechnical Commision

IMEI - International Mobile Station Equipment Identity

IPC - Inter-Process Communication, communication between two tasks running in different processes.

IRQ - Interrupt Request

ISR - Interrupt Service Routine

IWC - Inter-World Communication, communication between two tasks running in different worlds.

J

JNI - Java Native Interface

K

KiB - Kibibyte, IEC term for 1024 Byte. The commonly used KB actually refers to 1.000 Byte.

M

MC - MobiCore

MCI - MobiCore Control Interface

MCP - MobiCore Control Protocol

MiB - Mebibyte, IEC term for 1024 KiB. The commonly used MB actually refers to 1.000 KB.

MU - Mapping Unit. A Trustlet hat an address space of 8 MiB, where each MiB is a mapping unit. TCI/DCI is in the 2nd MU, 1 MiB WSM Blocks can be mapped in the 3rd to 8th MU

MMU - Memory Mapping Unit

MPU - Memory Protection Unit

MobiCore - G&D MobiCore

MobiCore System - The MobiCore System is an instantiation of a TEE as defined in OMTP. It is the software system consisting of a MobiCore enabled device and invoked by external components.

N

NQ - Notification Queue

NQ-IRQ - Notification Queue Interrupt raised by the Mobicore

S-SIQ - Secure Simulated Interrupt (becomes a IRQ for NWd)

N-SIQ - Non-secure Simulated Interrupt triggered by parameterized SMC call to Monitor from NWd

NWd - Normal World (the term "non-secure world" should be avoided). The software system running on an ARM TrustZone enabled device in non secure mode.

O

OTA - Over-the-air

OS - Operating System

P

PID - Process ID

PA - Provisioning Agent, NWd counterpart of the SWd Content Management Trustlet and cooperates with that System Trustlet for Content Management operations. The Provisioning Agent communicates with the Sytem Owner / Root and Service Provider backends. Furthermore the Provisoning Agent is responsible to store Content Management Secure Objects in NWd memory space.

R

RFU - Reserved for future use

Rich OS - The platform OS like Symbian, Android, iOS, ... running in the NWd.

RISC - Reduced Instcruction Set Computer

Root Container - The Root Container is associated to the role System Owner / Root and is available after MobiCore Root Registration. It organizes the Service Provider Containers of the Service Providers registered at the device.

RPC - Remote Procedure Call

RVCT - ARM RealView Compilation Tools, now called ARM Compiler

RVDS - ARM RealView Development Suite, discontinued and now replaced by DS-5

S

SE - Secure Element

SO - Secure Object, used for the persistent storage of Authentication Token, Root, Service Provider, Trustlet Container and Trustlet Personalization data. A Secure Object is an encrypted and integrity protected data package of that data. A Secure Object is transferred from SWd to the Provisioning Agent Client application to be persisted in a NWd storage space.

Service - A Service consists of an Application, at least one Trustlet and its TLC.

Service Provider Container - A Service Provider Container is associated to a specific Service Provider holding content management relevant Service Provider data. A Service Provider Container is available after the registration of the Service Provider to the device. The Service Provider Container organizes the Trustlet Containers of its Trustlets.

SiP - Silicon Provider

SMC - Secure Monitor Call, instruction of the ARM CPU

SP - Service Provider

SP_ID - Service Provider ID used to identify a registered Service Provider in MobiCore. SP_ID is part of the Service Provider Container.

SUID - SoC Unique Identifier. The SUID is an identifier of the SoC and is unique within all SoCs. As the SoC is build in a device (handset) the SUID is used to identify the device in the MobiCore ecosystem.

SoC - System on a Chip

SWd - Secure World, software system running in secure mode in the ARM TrustZone. MobiCore is running in the SWd.

T

TCI - Trustlet Control Interface.

TEE - Trusted Execution Environment. MobiCore is an instantiation of a TEE. The term TEE is defined in the OMTP specifications.

TL - Trustlet, Application running on MobiCore

TLAPI - Trustlet API

TLC - Trustlet Connector. Software running in the NWd providing a high level convenience interface to access a Trustlet in the SWd by the NWd client.

Trustlet - Trusted Application on the SWd side executing specific security services in the MobiCore runtime environment. The security services provided by a Trustlet are called by NWd client applications. MobiCore differentiates between System Trustlets like CM Trustlet and Trustlets of Service Providers.

Trustlet Container - A Trustlet Container is a container holding content management relevant data of a Trustlet (e.g. UUID and the Service Provider's key to encrypt Trustlet code).

TZ - ARM TrustZone

TZPC - TrustZone Protection Controller. The TZPC controls the configuration of memory regions and peripherals to be secure or non-secure.

U

UUID - Universal Unique Identifier. The UUID is an identifier standard used in software construction, standardized by the Open Software Foundation (OSF) as part of the Distributed Computing Environment (DCE). The UUID in the MobiCore ecosystem is used to uniquely identify Trustlets.

W

WSM - World Shared Memory. Shared memory which will be used for IPC between NWd and SWd.

Chapter 2

Module Index

2.1 Modules

Here is a list of all modules:

Chapter 3

Data Structure Index

3.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 4

File Index

4.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 5

Module Documentation

5.1 Test Subjects

The following test subjects are part of the MobiCore Integration Tests.

Modules

- Performance

  This subject contains the test cases for performance testing.

- Mobicore Inter Process Call Interface

  These tests bypass the Trustlet API and issue IPCs directly.

- MemoryManagement

  These tests verify the correct working of allocations and deallocations in NWd components,
  as well as correct setup of shared memory with SWd.

- McDrvApi

  The MobiCore Driver API functions are tested here.

- Runtime

  Test of the runtime behavior of SWd components.

- Trustlet API (TIApi)

  These tests contain good cases and bad cases for the API functions.

5.1.1 Detailed Description

The following test subjects are part of the MobiCore Integration Tests.

5.2 Performance

This subject contains the test cases for performance testing.

This subject contains the test cases for performance testing.

5.3 Inter-world communication functions [COM].

5.4 Security support functions [SEC].

5.5 MobiCore system functions [SYS].

5.6 Mobicore Inter Process Call Interface

These tests bypass the Trustlet API and issue IPCs directly.

These tests bypass the Trustlet API and issue IPCs directly. All existing MobiCore threads are contacted. Because Trustlets are isolated, only IPCH should respond. Drivers and other Trustlets should not be reachable. RTM threads other than IPCH should not interact with Trustlet, e.g. ignore message, or send error message.

5.7 Precondition setup

Test groups defined by target set up and and tear down.

Modules

- Standard session opened

5.7.1 Detailed Description

Test groups defined by target set up and and tear down.

5.8 Standard session opened

5.9 MemoryManagement

These tests verify the correct working of allocations and deallocations in NWd components,

as well as correct setup of shared memory with SWd.

Modules

- 00100_mcMapBoundaryTests

  Involved Components and their required Functionality

- Example Sub Group - TCI Buffer Tests

5.9.1 Detailed Description

These tests verify the correct working of allocations and deallocations in NWd components,

as well as correct setup of shared memory with SWd.

5.10 00100_mcMapBoundaryTests

Involved Components and their required Functionality

Functions

- TEST_SCENARIO (00100_mcMapBoundaryTests)
- TestCase_GC00100_CheckBufferBoundariesSize1 ()

  Verifies that the bidirectional NWd, SWd read write operations are successful on the buffer and page boundaries for a buffer size which is smaller that 1 page.

- TestCase_GC00101_CheckBufferBoundariesSize1Page ()
- TestCase_GC00102_CheckBufferBoundariesSize2Pages ()
- TestCase_GC00103_CheckBufferBoundariesSize3Pages ()
- TestCase_GC00104_CheckBufferBoundariesSize4Pages ()
- TestCase_GC00105_CheckBufferBoundariesSize8Pages ()
- TestCase_GC00106_CheckBufferFillAndReadbackSize1 ()
- TestCase_GC00107_CheckBufferFillAndReadbackSize1Page ()
- TestCase_GC00108_CheckBufferFillAndReadbackSize2Pages ()
- TestCase_GC00109_CheckBufferFillAndReadbackSize3Pages ()
- TestCase_GC00110_CheckBufferFillAndReadbackSize4Pages ()
- TestCase_GC00111_CheckBufferFillAndReadbackSize8Pages ()
- TestCase_GC00112_MapAllAndReadbackSize1 ()
- TestCase_GC00113_MapAllAndReadbackSize1Page ()
- TestCase_GC00114_MapAllAndReadbackSize2Pages ()
- TestCase_GC00115_MapAllAndReadbackSize3Pages ()
- TestCase_GC00116_MapAllAndReadbackSize4Pages ()
- TestCase_GC00117_MapAllAndReadbackSize8Pages ()
- TestCase_GC00118_MapAllAndReadbackSize255Pages ()
- TestCase_GC00119_MapDoesntClear ()
- TestCase_BC00120_TrustletPagefaultOnWrongAddress ()

5.10.1 Detailed Description

Involved Components and their required Functionality

1. ClientLib
   (a) Can connect to daemon
   (b) Can reserve memory for connection
2. McDriverDaemon
   (a) Daemon running
   (b) Daemon socket accessible by TLC
   (c) mcOpenDevice()
3. McDriverModule
   (a) Module loaded
   (b) Module socket accessible by daemon
   (c) Module listens for connection
4. Trustlet TlTestMemoryManagement
   (a) Trustlet copied to Android on Test device in /data/app/ as 05010000000000000000000000000000.trst
   (b) Trustlet runnable and bug-free

Generic preconditions (setup)

Generic Preconditions are the same for every test case. They are executed always in front of every single test case.

- open device using the default ID by calling mcOpenDevice()
- allocate a WSM buffer (4k)
- open a session to the Trustlet

Generic post processing (teardown)

Generic post processing is the same for every test case. It is executed always after every single test case.

- close session to the Trustlet
- free allocated memory of WSM buffer
- close Device by calling mcCloseDevice()

5.10.2 Function Documentation

5.10.2.1 TEST_SCENARIO (00100_mcMapBoundaryTests)

Description:

TLC maps additional Bulk memory to Trustlet and tests the access via TLC and Trustlet
Testsequence:
1. TLC fills Bulk buffer with a number and notifies the Trustlet
2. TL reads number from the Bulk buffer and increments it
3. TLC reads back number after notification from the TL

Precondition

Involved components running and functioning

Verification:

returns MC_DRV_OK.
Test output is OK.

5.10.2.2 TestCase_GC00100_CheckBufferBoundariesSize1 ( )

Verifies that the bidirectional NWd, SWd read write operations are successful

on the buffer and page boundaries for a buffer size which is smaller that 1 page.

Description:

Verifies that the bidirectional NWd, SWd read write operations are successful
on the buffer and page boundaries for a buffer size which is smaller than 1 page.

Precondition

Involved components running and functioning

Focus:

CMD_TEST_FILL_MEM command is executable for single byte memory insertion

Verification:

Page and buffer boundaries are written from SWd to NWd and verified
Data is written from NWd to SWd and verified. CMD_TEST_FILL_MEM and CMD_-TEST_COMPARE_MEM commands are successfuly executed.

Postcondition

Allocated fields are freed.

Test ID:

MemoryManagement_00100_mcMapBoundaryTests_GC00100_CheckBufferBoundariesSize1

5.10.2.3 TestCase_GC00101_CheckBufferBoundariesSize1Page ( )

Description:

Verifies that the bidirectional NWd, SWd read write operations are successful on the buffer and page boundaries for a buffer sized smaller than 2 pages. Proves that memory allocation does not cause a boundary related problems with 2 pages affected.

Precondition

Involved components running and functioning

Focus:

CMD_TEST_FILL_MEM command is executable for over 1 page size memory insertion

Verification:

Page and buffer boundaries are written from SWd to NWd and verified
Data is written from NWd to SWd and verified.
CMD_TEST_FILL_MEM and CMD_TEST_COMPARE_MEM commands are successfuly executed.

Postcondition

Allocated fields are freed.

Test ID:

MemoryManagement_00100_mcMapBoundaryTests_GC00101_CheckBufferBoundariesSize1Page

5.10.2.4 TestCase_GC00102_CheckBufferBoundariesSize2Pages ( )

Description:

Verifies that the bidirectional NWd, SWd read write operations are successful on the buffer and page boundaries for a buffer sized smaller than 3 pages.

Precondition

Involved components running and functioning

Focus:

CMD_TEST_FILL_MEM command is executable for over 2 page size memory insertion

Verification:

Page and buffer boundaries are written from SWd to NWd and verified
Data is written from NWd to SWd and verified. CMD_TEST_FILL_MEM and CMD_-TEST_COMPARE_MEM commands are successfuly executed.

Postcondition

Allocated fields are freed.

Test ID:

MemoryManagement_00100_mcMapBoundaryTests_GC00102_CheckBufferBoundariesSize2Pages

5.10.2.5 TestCase_GC00103_CheckBufferBoundariesSize3Pages ( )

Description:

Verifies that the bidirectional NWd, SWd read write operations are successful on the buffer and page boundaries for a buffer sized smaller than 4 pages.

Precondition

Involved components running and functioning

Focus:

CMD_TEST_FILL_MEM command is executable for over 3 page size memory insertion

Verification:

Page and buffer boundaries are written from SWd to NWd and verified
Data is written from NWd to SWd and verified. CMD_TEST_FILL_MEM and CMD_TEST_COMPARE_MEM commands are successfuly executed.

Postcondition

Allocated fields are freed.

Test ID:

MemoryManagement_00100_mcMapBoundaryTests_GC00103_CheckBufferBoundariesSize3Pages

5.10.2.6 TestCase_GC00104_CheckBufferBoundariesSize4Pages ( )

Description:

Verifies that the bidirectional NWd, SWd read write operations are successful on the buffer and page boundaries for a buffer sized smaller than 5 pages.

Precondition

Involved components running and functioning

Focus:

CMD_TEST_FILL_MEM command is executable for over 4 page size memory insertion

Verification:

Page and buffer boundaries are written from SWd to NWd and verified
Data is written from NWd to SWd and verified.
CMD_TEST_FILL_MEM and CMD_TEST_COMPARE_MEM commands are successfuly executed.

Postcondition

Allocated fields are freed.

Test ID:

MemoryManagement_00100_mcMapBoundaryTests_GC00104_CheckBufferBoundariesSize4Pages

5.10.2.7 TestCase_GC00105_CheckBufferBoundariesSize8Pages ( )

Description:

Verifies that the bidirectional NWd, SWd read write operations are successful on the buffer and page boundaries for a buffer sized smaller than 9 pages.

Precondition

Involved components running and functioning

Focus:

CMD_TEST_FILL_MEM command is executable for over 8 page size memory insertion

Verification:

Page and buffer boundaries are written from SWd to NWd and verified
Data is written from NWd to SWd and verified. CMD_TEST_FILL_MEM and CMD_-TEST_COMPARE_MEM commands are successfuly executed.

Postcondition

Allocated fields are freed.

Test ID:

MemoryManagement_00100_mcMapBoundaryTests_GC00105_CheckBufferBoundariesSize8Pages

5.10.2.8 TestCase_GC00106_CheckBufferFillAndReadbackSize1 ( )

Description:

Precondition

Involved components running and functioning

Focus:

mcMap call maps the SWd memory synchronizes with NWd buffer.

Verification:

Data block is written to the NWd via SWd.
Data is written from NWd to SWd and verified.
CMD_TEST_FILL_MEM and CMD_TEST_COMPARE_MEM commands are successfuly executed.

Postcondition

Bindings are unMap'ped and allocated fields are freed.

Test ID:

MemoryManagement_00100_mcMapBoundaryTests_GC00106_CheckBufferFillAndReadbackSize1

5.10.2.9 TestCase_GC00107_CheckBufferFillAndReadbackSize1Page ( )

Description:

Verifies that the integrity of randomly generated data written
into the NWd via SWd is kept totally. Data consistency check is tested for data
larger than a page size.

Precondition

Involved components running and functioning

Focus:

mcMap call maps the SWd memory synchronizes with NWd buffer.

Verification:

Data block which is larger than a page size is written to the NWd via SWd.
Data is written from NWd to SWd and verified.
CMD_TEST_FILL_MEM and CMD_TEST_COMPARE_MEM commands are successfuly executed.

Postcondition

Allocated fields are freed.

Test ID:

MemoryManagement_00100_mcMapBoundaryTests_GC00107_CheckBufferFillAndReadbackSize1Page

5.10.2.10 TestCase_GC00108_CheckBufferFillAndReadbackSize2Pages ( )

Description:

Precondition

Involved components running and functioning

Focus:

mcMap call maps the SWd memory synchronizes with NWd buffer.

Verification:

Data block which is larger than 2 pages' size is written to the NWd via SWd.
Data is written from NWd to SWd and verified.
CMD_TEST_FILL_MEM and CMD_TEST_COMPARE_MEM commands are successfuly executed.

Postcondition

Allocated fields are freed.

Test ID:

MemoryManagement_00100_mcMapBoundaryTests_GC00108_CheckBufferFillAndReadbackSize2Pages

5.10.2.11 TestCase_GC00109_CheckBufferFillAndReadbackSize3Pages ( )

Description:

Precondition

Involved components running and functioning

Focus:

mcMap call maps the SWd memory synchronizes with NWd buffer.

Verification:

Data block which is larger than 3 pages' size is written to the NWd via SWd.
Data is written from NWd to SWd and verified.
CMD_TEST_FILL_MEM and CMD_TEST_COMPARE_MEM commands are successfuly executed.

Postcondition

Allocated fields are freed.

Test ID:

MemoryManagement_00100_mcMapBoundaryTests_GC00109_CheckBufferFillAndReadbackSize3Pages

5.10.2.12 TestCase_GC00110_CheckBufferFillAndReadbackSize4Pages ( )

Description:

Precondition

Involved components running and functioning

Focus:

mcMap call maps the SWd memory synchronizes with NWd buffer.

Verification:

Data block which is larger than 4 pages' size is written to the NWd via SWd.
Data is written from NWd to SWd and verified.
CMD_TEST_FILL_MEM and CMD_TEST_COMPARE_MEM commands are successfuly executed.

Postcondition

Allocated fields are freed.

Test ID:

MemoryManagement_00100_mcMapBoundaryTests_GC00110_CheckBufferFillAndReadbackSize4Pages

5.10.2.13 TestCase_GC00111_CheckBufferFillAndReadbackSize8Pages ( )

Description:

Precondition

Involved components running and functioning

Focus:

mcMap call maps the SWd memory synchronizes with NWd buffer.

Verification:

Data block which is larger than 8 pages' size is written to the NWd via SWd.
Data is written from NWd to SWd and verified.
CMD_TEST_FILL_MEM and CMD_TEST_COMPARE_MEM commands are successfuly executed.

Postcondition

Allocated fields are freed.

Test ID:

MemoryManagement_00100_mcMapBoundaryTests_GC00111_CheckBufferFillAndReadbackSize8Pages

5.10.2.14 TestCase_GC00112_MapAllAndReadbackSize1 ( )

Description:

Verifies the SWd to NWd memory access with multiple buffers allocated, read and written.
allocates 4 buffers with 1 byte, writes the randomly generated info to NWd memory via SWd and verifies the data.

Precondition

Involved components running and functioning

Focus:

mcMap call maps the SWd memory synchronizes with NWd buffer.

Verification:

4 identical data blocks which sized 1 byte is written to the NWd via SWd.
Data is written from NWd to SWd and verified.
CMD_TEST_FILL_MEM and CMD_TEST_COMPARE_MEM commands are successfuly executed.

Postcondition

Allocated fields are freed.

Test ID:

MemoryManagement_00100_mcMapBoundaryTests_GC00112_MapAllAndReadbackSize1

5.10.2.15 TestCase_GC00113_MapAllAndReadbackSize1Page ( )

Description:

Verifies the SWd to NWd memory access with multiple buffers allocated, read and written.
allocates 4 buffers with data sized larger than 1 page size, writes the randomly generated info to
NWd memory via SWd and verifies the data.

Precondition

Involved components running and functioning

Focus:

mcMap call maps the SWd memory synchronizes with NWd buffer.

Verification:

4 identical data blocks which are sized larger than 1 pages is written to the NWd via SWd.

Data is written from NWd to SWd and verified.
CMD_TEST_FILL_MEM and CMD_TEST_COMPARE_MEM commands are successfuly executed.

Postcondition

Allocated fields are freed.

Test ID:

MemoryManagement_00100_mcMapBoundaryTests_GC00113_MapAllAndReadbackSize1Page

5.10.2.16 TestCase_GC00114_MapAllAndReadbackSize2Pages ( )

Description:

Verifies the SWd to NWd memory access with multiple buffers allocated, read and written.
allocates 4 buffers with data sized larger than 2 pages' size, writes the randomly generated info to
NWd memory via SWd and verifies the data.

Precondition

Involved components running and functioning

Focus:

mcMap call maps the SWd memory synchronizes with NWd buffer.

Verification:

4 identical data blocks which are sized larger than 2 pages is written to the NWd via SWd.
Data is written from NWd to SWd and verified.
CMD_TEST_FILL_MEM and CMD_TEST_COMPARE_MEM commands are successfuly executed.

Postcondition

Allocated fields are freed.

Test ID:

MemoryManagement_00100_mcMapBoundaryTests_GC00114_MapAllAndReadbackSize2Pages

5.10.2.17 TestCase_GC00115_MapAllAndReadbackSize3Pages ( )

Description:

Verifies the SWd to NWd memory access with multiple buffers allocated, read and written.

allocates 4 buffers with data sized larger than 3 pages' size, writes the randomly generated info to
NWd memory via SWd and verifies the data.

Precondition

Involved components running and functioning

Focus:

mcMap call maps the SWd memory synchronizes with NWd buffer.

Verification:

4 identical data blocks which are sized larger than 3 pages is written to the NWd via SWd.
Data is written from NWd to SWd and verified.
CMD_TEST_FILL_MEM and CMD_TEST_COMPARE_MEM commands are successfuly executed.

Postcondition

Allocated fields are freed.

Test ID:

MemoryManagement_00100_mcMapBoundaryTests_GC00115_MapAllAndReadbackSize3Pages

5.10.2.18 TestCase_GC00116_MapAllAndReadbackSize4Pages ( )

Description:

Verifies the SWd to NWd memory access with multiple buffers allocated, read and written.
allocates 4 buffers with data sized larger than 4 pages' size, writes the randomly generated info to
NWd memory via SWd and verifies the data.

Precondition

Involved components running and functioning

Focus:

mcMap call maps the SWd memory synchronizes with NWd buffer.

Verification:

4 identical data blocks which are sized larger than 4 pages is written to the NWd via SWd.
Data is written from NWd to SWd and verified.
CMD_TEST_FILL_MEM and CMD_TEST_COMPARE_MEM commands are successfuly executed.

Postcondition

Allocated fields are freed.

Test ID:

MemoryManagement_00100_mcMapBoundaryTests_GC00116_MapAllAndReadbackSize4Pages

5.10.2.19 TestCase_GC00117_MapAllAndReadbackSize8Pages ( )

Description:

Verifies the SWd to NWd memory access with multiple buffers allocated, read and written.
allocates 4 buffers with data sized larger than 8 pages' size, writes the randomly generated info to
NWd memory via SWd and verifies the data.

Precondition

Involved components running and functioning

Focus:

mcMap call maps the SWd memory synchronizes with NWd buffer.

Verification:

4 identical data blocks which are sized larger than 8 pages is written to the NWd via SWd.
Data is written from NWd to SWd and verified.
CMD_TEST_FILL_MEM and CMD_TEST_COMPARE_MEM commands are successfuly executed.

Postcondition

Allocated fields are freed.

Test ID:

MemoryManagement_00100_mcMapBoundaryTests_GC00117_MapAllAndReadbackSize8Pages

5.10.2.20 TestCase_GC00118_MapAllAndReadbackSize255Pages ( )

Description:

Verifies the SWd to NWd memory access with multiple buffers allocated, read and written.
allocates 4 buffers with data sized 255 pages' size, writes the randomly generated info to
NWd memory via SWd and verifies the data.

Precondition

Involved components running and functioning

Focus:

mcMap call maps the SWd memory synchronizes with NWd buffer.

Verification:

4 identical data blocks which are sized as big as 255 pages is written to the NWd via SWd.
Data is written from NWd to SWd and verified.
CMD_TEST_FILL_MEM and CMD_TEST_COMPARE_MEM commands are successfuly executed.

Postcondition

Allocated fields are freed.

Test ID:

MemoryManagement_00100_mcMapBoundaryTests_GC00118_MapAllAndReadbackSize255Pages

5.10.2.21 TestCase_GC00119_MapDoesntClear ( )

Description:

TLC writes something in buffer and then maps the buffer to Trustlet
right after the moment of mapping buffer is verified to be cleared totally.

Precondition

Involved components running and functioning

Focus:

mcMap clears the buffer when mapping takes place

Verification:

Mapping does not clear the buffer
Test output is OK.

Postcondition

Allocated fields are freed.

Test ID:

MemoryManagement_00100_mcMapBoundaryTests_GC00119_MapDoesntClear

5.10.2.22 TestCase_BC00120_TrustletPagefaultOnWrongAddress ( )

Description:

TLC maps a buffer to Trustlet and then orders Trustlet to write next to the buffer.

Note

This is more a test of mcUnmap(), bad we keep it here because it resembles the other tests here. TODO: move test in separate file or to unmap scenario

Precondition

Involved components running and functioning

Focus:

mcNotify informs TL to swtich the access offset in the buffer.

Verification:

Trustlet pagefaults
mcUnmap() returns MC_DRV_OK, even if trustlet is dead. Test output is OK.

Postcondition

Test ID:

MemoryManagement_00100_mcMapBoundaryTests_BC00120_TrustletPagefaultOnWrongAddress

5.11 Example Sub Group - TCI Buffer Tests

5.12 McDrvApi

The MobiCore Driver API functions are tested here.

Modules

- 00100_mcOpenDevice

  Test Scenario for McDriverAPI function mcOpenDevice()

- 00200_mcCloseDevice

  Test Scenario for McDriverAPI function mcCloseDevice()

- 00300_mcMallocWsm

  Test Scenario for McDriverAPI function mcMallocWsm()

- 00400_mcFreeWsm

  Involved Components and their required Functionality

- 00500_mcOpenSession

  Involved Components and their required Functionality

- 00600_mcCloseSession

  Involved Components and their required Functionality

- 00700_mcNotify

  Involved Components and their required Functionality

- 00800_mcWaitNotification

  Involved Components and their required Functionality

- 00900_mcGetSessionErrorCode

  Involved Components and their required Functionality

- 01000_mcMap

  Involved Components and their required Functionality

- 01100_mcUnmap

  Involved Components and their required Functionality

- 01200_mcGetRndSeed

  Involved Components and their required Functionality

5.12.1 Detailed Description

The MobiCore Driver API functions are tested here. Testcases in MobiCore Driver API also test the communication API [COM] of the Trustlet API by using tlNotify and tlWaitNotification.

Introduction

Tests checking the basic functionality of the MobiCore driver interface.

5.13 00100_mcOpenDevice

Test Scenario for McDriverAPI function mcOpenDevice()

Functions

- TestCase_GC00100_IdDefault ()
- TestCase_BC00101_InvalidDevice1 ()
- TestCase_BC00102_InvalidDevice2 ()
- TestCase_BC00103_InvalidDeviceMinus1 ()
- TestCase_BC00104_OpenRepeatedly ()

5.13.1 Detailed Description

Test Scenario for McDriverAPI function mcOpenDevice()

Involved Components and their required Functionality

1. ClientLib
   (a) Can connect to daemon
   (b) Can reserve memory for connection
2. McDriverDaemon
   (a) Daemon running
   (b) Daemon socket accessible by TLC
3. McDriverModule
   (a) Module loaded
   (b) Module socket accessible by daemon
   (c) Module listens for connection

Generic preconditions (setup)

Generic Preconditions are the same for every test case. They are executed always in front of every single test case.

- none

Generic post processing (teardown)

Generic post processing is the same for every test case. It is executed always after every single test case.

- close Device by calling mcCloseDevice()

5.13.2 Function Documentation

5.13.2.1 TestCase_GC00100_IdDefault ( )

Description:

TLC opens a new default device (MC_DEVICE_ID_DEFAULT)

Precondition

-

Focus:

mcOpenDevice(MC_DEVICE_ID_DEFAULT)

Verification:

MC_DRV_OK is returned

Postcondition

-

Test ID:

McDrvApi_00100_mcOpenDevice_GC00100_IdDefault

5.13.2.2 TestCase_BC00101_InvalidDevice1 ( )

Description:

TLC opens an invalid device (1)

Precondition

-

Focus:

mcOpenDevice(1)

Verification:

MC_DRV_ERR_UNKNOWN_DEVICE is returned

Postcondition

-

Test ID:

McDrvApi_00100_mcOpenDevice_BC00101_InvalidDevice1

5.13.2.3 TestCase_BC00102_InvalidDevice2 ( )

Description:

TLC opens an invalid device (2)

Precondition

-

Focus:

mcOpenDevice(2)

Verification:

MC_DRV_ERR_UNKNOWN_DEVICE is returned

Postcondition

-

Test ID:

McDrvApi_00100_mcOpenDevice_BC00102_InvalidDevice2

5.13.2.4 TestCase_BC00103_InvalidDeviceMinus1 ( )

Description:

TLC opens an invalid device (-1)

Precondition

-

Focus:

mcOpenDevice(-1)

Verification:

MC_DRV_ERR_UNKNOWN_DEVICE is returned

Postcondition

-

Test ID:

McDrvApi_00100_mcOpenDevice_BC00103_InvalidDeviceMinus1

5.13.2.5 TestCase_BC00104_OpenRepeatedly ( )

Description:

TLC opens same device a second time

Precondition

1. mcOpenDevice(MC_DEVICE_ID_DEFAULT) => returns MC_DRV_OK

Focus:

2. mcOpenDevice(MC_DEVICE_ID_DEFAULT)

Verification:

MC_DRV_ERR_INVALID_OPERATION is returned for second time

Postcondition

-

Test ID:

McDrvApi_00100_mcOpenDevice_BC00104_OpenRepeatedly

5.14 00200_mcCloseDevice

Test Scenario for McDriverAPI function mcCloseDevice()

Functions

- TestCase_GC00100_IdDefault ()
- TestCase_BC00101_InvalidDevice1 ()
- TestCase_BC00102_InvalidDevice2 ()
- TestCase_BC00103_InvalidDeviceMinus1 ()
- TestCase_BC00103_CloseRepeatedly ()

5.14.1 Detailed Description

Test Scenario for McDriverAPI function mcCloseDevice()

Involved Components and their required Functionality

1. ClientLib
   (a) Can connect to daemon
   (b) Can reserve memory for connection
2. McDriverDaemon

(a) Daemon running

(b) Daemon socket accessible by TLC

(c) mcOpenDevice()

3. McDriverModule

(a) Module loaded

(b) Module socket accessible by daemon

(c) Module listens for connection

Generic preconditions (setup)

Generic Preconditions are the same for every test case. They are executed always in front of every single test case.

- open device using the default ID by calling mcOpenDevice()

Generic post processing (teardown)

Generic post processing is the same for every test case. It is executed always after every single test case.

- close Device by calling mcCloseDevice() if it is not already closed

5.14.2 Function Documentation

5.14.2.1 TestCase_GC00100_IdDefault ( )

Description:

TLC closes a device session using the default device ID (MC_DEVICE_ID_DEFAULT)

Precondition

A device has been opened using the default ID in Generic preconditions (setup)

Focus:

mcCloseDevice(MC_DEVICE_ID_DEFAULT)

Verification:

MC_DRV_OK is returned

Postcondition

-

Test ID:

McDrvApi_00200_mcCloseDevice_GC00100_IdDefault

5.14.2.2 TestCase_BC00101_InvalidDevice1 ( )

Description:

TLC closes a device with an invalid device ID (1)

Precondition

A device has been opened using the default ID in Generic preconditions (setup)

Focus:

mcCloseDevice(1)

Verification:

MC_DRV_ERR_UNKNOWN_DEVICE is returned

Postcondition

-

Test ID:

McDrvApi_00200_mcCloseDevice_BC00101_InvalidDevice1

5.14.2.3 TestCase_BC00102_InvalidDevice2 ( )

Description:

TLC closes a device with an invalid device ID (2)

Precondition

A device has been opened using the default ID in Generic preconditions (setup)

Focus:

mcCloseDevice(2)

Verification:

MC_DRV_ERR_UNKNOWN_DEVICE is returned

Postcondition

-

Test ID:

McDrvApi_00200_mcCloseDevice_BC00102_InvalidDevice2

5.14.2.4 TestCase_BC00103_InvalidDeviceMinus1 ( )

Description:

TLC closes a device with an invalid device ID (-1)

Precondition

A device has been opened using the default ID in Generic preconditions (setup)

Focus:

mcOpenDevice(-1)

Verification:

MC_DRV_ERR_UNKNOWN_DEVICE is returned

Postcondition

-

Test ID:

McDrvApi_00200_mcCloseDevice_BC00103_InvalidDeviceMinus1

5.14.2.5 TestCase_BC00103_CloseRepeatedly ( )

Description:

TLC closes a device which is already closed

Precondition

A device has been opened using the default ID in Generic preconditions (setup)
1. mcCloseDevice(MC_DEVICE_ID_DEFAULT) => returns MC_DRV_OK

Focus:

2. mcCloseDevice(MC_DEVICE_ID_DEFAULT)

Verification:

MC_DRV_ERR_UNKNOWN_DEVICE is returned for second time

Postcondition

-

Test ID:

McDrvApi_00200_mcCloseDevice_BC00103_CloseRepeatedly

5.15 00300_mcMallocWsm

Test Scenario for McDriverAPI function mcMallocWsm()

Functions

- TestCase_GC00100_IdDefault4k ()
- TestCase_BC00101_InvalidDevice1 ()
- TestCase_BC00101_InvalidDevice4 ()
- TestCase_BC00102_InvalidDeviceMinus1 ()
- TestCase_BC00103_Len0 ()
- TestCase_BC00104_LenMinus1 ()
- TestCase_BC00105_WsmNullPointer ()
- TestCase_BC00106_ClosedDevice ()
- TestCase_GC00107_StresstestMultipleAllocations ()
- TestCase_GC00108_StresstestMultipleSizes ()

5.15.1 Detailed Description

Test Scenario for McDriverAPI function mcMallocWsm()

Involved Components and their required Functionality

1. ClientLib
   (a) Can connect to daemon
   (b) Can reserve memory for connection
2. McDriverDaemon
   (a) Daemon running
   (b) Daemon socket accessible by TLC
   (c) mcOpenDevice()
3. McDriverModule
   (a) Module loaded
   (b) Module socket accessible by daemon
   (c) Module listens for connection

Generic preconditions (setup)

Generic Preconditions are the same for every test case. They are executed always in front of every single test case.

- open device using the default ID by calling mcOpenDevice()

Generic post processing (teardown)

Generic post processing is the same for every test case. It is executed always after every single test case.

- free allocated memory by calling mcFreeWsm() if it has been allocated
- close Device by calling mcCloseDevice()

5.15.2 Function Documentation

5.15.2.1 TestCase_GC00100_IdDefault4k ( )

Description:

TLC allocates 4kB of WSM

Precondition

A device has been opened using the default ID in Generic preconditions (setup)

Focus:

mcMallocWsm(MC_DEVICE_ID_DEFAULT, 0, 4096, &tci, 0)

Verification:

MC_DRV_OK is returned

Postcondition

-

Test ID:

McDrvApi_00300_mcMallocWsm_GC00100_IdDefault4k

5.15.2.2 TestCase_BC00101_InvalidDevice1 ( )

Description:

TLC calls mcMallocWsm with an invalid device id (1)

Precondition

A device has been opened using the default ID in Generic preconditions (setup)

Focus:

mcMallocWsm(1, 0, 4096, &tci, 0)

Verification:

MC_DRV_ERR_UNKNOWN_DEVICE is returned

Postcondition

-

Test ID:

McDrvApi_00300_mcMallocWsm_BC00101_InvalidDevice1

5.15.2.3 TestCase_BC00101_InvalidDevice4 ( )

Description:

TLC calls mcMallocWsm with an invalid device id (4)

Precondition

A device has been opened using the default ID in Generic preconditions (setup)

Focus:

mcMallocWsm(4, 0, 4096, &tci, 0)

Verification:

MC_DRV_ERR_UNKNOWN_DEVICE is returned

Postcondition

-

Test ID:

McDrvApi_00300_mcMallocWsm_BC00101_InvalidDevice4

5.15.2.4 TestCase_BC00102_InvalidDeviceMinus1 ( )

Description:

TLC calls mcMallocWsm with an invalid device id (-1)

Precondition

A device has been opened using the default ID in Generic preconditions (setup)

Focus:

mcMallocWsm(-1, 0, 4096, &tci, 0)

Verification:

MC_DRV_ERR_UNKNOWN_DEVICE is returned

Postcondition

-

Test ID:

McDrvApi_00300_mcMallocWsm_BC00102_InvalidDeviceMinus1

5.15.2.5 TestCase_BC00103_Len0 ( )

Description:

TLC calls mcMallocWsm with an unsupported len parameter (0)

Precondition

A device has been opened using the default ID in Generic preconditions (setup)

Focus:

mcMallocWsm(MC_DEVICE_ID_DEFAULT, 0, 0, &tci, 0)

Verification:

MC_DRV_ERR_NO_FREE_MEMORY is returned

Postcondition

-

Test ID:

McDrvApi_00300_mcMallocWsm_BC00103_Len0

5.15.2.6 TestCase_BC00104_LenMinus1 ( )

Description:

TLC calls mcMallocWsm with an unsupported len parameter (-1)

Precondition

A device has been opened using the default ID in Generic preconditions (setup)

Focus:

mcMallocWsm(MC_DEVICE_ID_DEFAULT, 0, -1, &tci, 0)

Verification:

MC_DRV_ERR_NO_FREE_MEMORY is returned

Postcondition

-

Test ID:

McDrvApi_00300_mcMallocWsm_BC00104_LenMinus1

5.15.2.7 TestCase_BC00105_WsmNullPointer ( )

Description:

TLC calls mcMallocWsm with a NULL pointer

Precondition

A device has been opened using the default ID in Generic preconditions (setup)

Focus:

mcMallocWsm(MC_DEVICE_ID_DEFAULT, 0, 4096, 0, 0)

Verification:

MC_DRV_ERR_INVALID_PARAMETER is returned

Postcondition

-

Test ID:

McDrvApi_00300_mcMallocWsm_BC00105_WsmNullPointer

5.15.2.8 TestCase_BC00106_ClosedDevice ( )

Description:

TLC calls mcMallocWsm with a not yet opened device

Precondition

Close device using the default ID

Focus:

mcMallocWsm

Verification:

MC_DRV_ERR_UNKNOWN_DEVICE is returned

Postcondition

Open device again using the default ID

Test ID:

McDrvApi_00300_mcMallocWsm_BC00106_ClosedDevice

5.15.2.9 TestCase_GC00107_StresstestMultipleAllocations ( )

Description:

TLC calls mcMallocWsm 16 times
After that, the memory (available to the TLC) is exhausted.
After that, the TLC frees the memory again.
Several runs of this test should not leak any other memory

Precondition

A device has been opened using the default ID in Generic preconditions (setup)

Focus:

call mcMallocWsm several times allocating blocks of 4kB

Verification:

MC_DRV_OK is returned until no more memory is available then
MC_DRV_ERR_NO_FREE_MEMORY is returned.

Postcondition

All memory is freed again by calling mcFreeWsm(), no memory leaks occur

Test ID:

McDrvApi_00300_mcMallocWsm_GC00107_StresstestMultipleAllocations

5.15.2.10 TestCase_GC00108_StresstestMultipleSizes ( )

Description:

TLC calls mcMallocWsm and iteratively increases the size.
The TLC frees the memory after each allocation.
Several runs of this test should not leak any other memory

Precondition

A device has been opened using the default ID in Generic preconditions (setup)

Focus:

call mcMallocWsm several times allocating blocks with increasing block size

Verification:

MC_DRV_OK is returned until no more memory is available then
MC_DRV_ERR_NO_FREE_MEMORY is returned.

Postcondition

memory is not freed explicitely, but device is closed by generic post processing

Test ID:

McDrvApi_00300_mcMallocWsm_GC00108_StresstestMultipleSizes

5.16 00400_mcFreeWsm

Involved Components and their required Functionality

Functions

- TestCase_GC00100_DeviceDefault () 1)
- TestCase_BC00101_DeviceInvalid () 1)
- TestCase_BC00102_WsmNullPointer () 1)
- TestCase_BC00103_MemoryNotManagedByLib () 1)
- TestCase_BC00104_OnClosedDevice () 2)

5.16.1 Detailed Description

Involved Components and their required Functionality

1. ClientLib
   (a) Can connect to daemon
   (b) Can reserve memory for connection
2. McDriverDaemon
   (a) Daemon running
   (b) Daemon socket accessible by TLC
   (c) mcOpenDevice()
3. McDriverModule
   (a) Module loaded
   (b) Module socket accessible by daemon
   (c) Module listens for connection **Generic preconditions (setup)**
   Generic Preconditions are the same for every test case. They are executed always in front of every single test case.

- open device using the default ID by calling mcOpenDevice()
- allocate a WSM buffer (4k)

Generic post processing (teardown)

Generic post processing is the same for every test case. It is executed always after every single test case.

- close session to the Trustlet
- free allocated memory of WSM buffer
- close Device by calling mcCloseDevice()

5.16.2 Function Documentation

5.16.2.1 TestCase_GC00100_DeviceDefault ( )

Description:

1. TLC allocates a block of WSM.
2. TLC frees the block again

Precondition

Involved components running and functioning

Focus:

mcFreeWsm works correctly.

Verification:

All checks pass, Test output is OK

Postcondition

Test ID:

McDrvApi_00400_mcFreeWsm_GC00100_DeviceDefault

5.16.2.2 TestCase_BC00101_DeviceInvalid ( )

Description:

TLC calls mcFreeeWsm with an unknown device id (1)

Precondition

Involved components running and functioning

Focus:

mcFreeWsm returns an error for an unknown device id.

Verification:

mcFreeWsm() returns MC_DRV_ERR_UNKNOWN_DEVICE, Test output is OK

Postcondition

Test ID:

McDrvApi_00400_mcFreeWsm_BC00101_DeviceInvalid

5.16.2.3 TestCase_BC00102_WsmNullPointer ( )

Description:

TLC calls mcFreeeWsm with a NULL pointer

Precondition

Involved components running and functioning

Focus:

mcFreeWsm returns an error for a null device id parameter.

Verification:

mcFreeWsm() returns MC_DRV_ERR_INVALID_PARAMETER, Test output is OK

Postcondition

Test ID:

McDrvApi_00400_mcFreeWsm_BC00102_WsmNullPointer

5.16.2.4 TestCase_BC00103_MemoryNotManagedByLib ( )

Description:

TLC calls mcFreeeWsm with memory not managed by MobiCore Driver

Precondition

Involved components running and functioning

Focus:

mcFreeWsm returns an error for a parameter points a memory block that is not controlled by driver.

Verification:

mcFreeWsm() returns MC_DRV_ERR_INVALID_PARAMETER, Test output is OK

Postcondition

Test ID:

McDrvApi_00400_mcFreeWsm_BC00103_MemoryNotManagedByLib

5.16.2.5 TestCase_BC00104_OnClosedDevice ( )

Description:

TLC calls mcFreeeWsm on a not yet opened device

Precondition

Involved components running and functioning

Focus:

mcFreeWsm returns an error for a parameter points an unknown device.

Verification:

mcFreeWsm() returns MC_DRV_ERR_UNKNOWN_DEVICE, Test output is OK

Postcondition

Test ID:

McDrvApi_00400_mcFreeWsm_BC00104_OnClosedDevice

5.17 00500_mcOpenSession

Involved Components and their required Functionality

Functions

- TestCase_GC00100_05010000000000000000000000000000 () 1)
- TestCase_BC00101_SessionNullPointer () 1)
- TestCase_BC00102_UuidNullPointer () 1)
- TestCase_BC00103_TciNullPointer () 1)
- TestCase_BC00104_LenBiggerMax () 1)
- TestCase_BC00105_LenMinus1 () 1)
- TestCase_BC00106_Len0 () 1)
- TestCase_BC00107_SessionhandleInvalid () 1)
- TestCase_BC00108_UuidUnknown1234123412341234 () 1)
- TestCase_GC00109_MaximumSessions () 2)
- TestCase_BC00110_CloseDeviceWithOpenSession () 1)
- TestCase_BC00111_LeaveSession () 3)

5.17.1 Detailed Description

Involved Components and their required Functionality

1. ClientLib
   (a) Can connect to daemon
   (b) Can reserve memory for connection
2. McDriverDaemon
   (a) Daemon running
   (b) Daemon socket accessible by TLC
   (c) mcOpenDevice()
3. McDriverModule
   (a) Module loaded
   (b) Module socket accessible by daemon
   (c) Module listens for connection
4. Trustlet TlIntegrationTest
   (a) Trustlet copied to Android on Test device in /data/app/ as 05010000000000000000000000000000.trst
   (b) Trustlet runnable and bug-free

Generic preconditions (setup)

Generic Preconditions are the same for every test case. They are executed always in front of every single test case.

- open device using the default ID by calling mcOpenDevice()
- allocate a WSM buffer (4k)
- open a session to the Trustlet

Generic post processing (teardown)

Generic post processing is the same for every test case. It is executed always after every single test case.

- close session to the Trustlet
- free allocated memory of WSM buffer
- close Device by calling mcCloseDevice()

5.17.2 Function Documentation

5.17.2.1 TestCase_GC00100_05010000000000000000000000000000 ( )

Description:

TLC opens a session to a trustlet

Precondition

Involved components running and functioning

Focus:

mcOpenSession opens a session successfully

Verification:

mcOpenSession() returns MC_DRV_OK,

Postcondition

sessionId is not zero and not -1,
Most significant bit in sessionId is not set,
Test output is OK

Test ID:

McDrvApi_00500_mcOpenSession_GC00100_05010000000000000000000000000000

5.17.2.2 TestCase_BC00101_SessionNullPointer ( )

Description:

TLC opens a session with NULL pointer parameter for session

Precondition

Involved components running and functioning

Focus:

mcOpenSession returns an error for a null session parameter.

Verification:

mcOpenSession() returns MC_DRV_ERR_INVALID_PARAMETER, Test output is OK

Postcondition

-

Test ID:

McDrvApi_00500_mcOpenSession_BC00101_SessionNullPointer

5.17.2.3 TestCase_BC00102_UuidNullPointer ( )

Description:

TLC opens a session with NULL pointer parameter for uuid

Precondition

Involved components running and functioning

Focus:

mcOpenSession returns an error for a null uuid parameter.

Verification:

mcOpenSession() returns MC_DRV_ERR_INVALID_PARAMETER, Test output is OK

Postcondition

-

Test ID:

McDrvApi_00500_mcOpenSession_BC00102_UuidNullPointer

5.17.2.4 TestCase_BC00103_TciNullPointer ( )

Description:

TLC opens a session with NULL pointer parameter for tci

Precondition

Involved components running and functioning

Focus:

mcOpenSession returns an error for a null tci parameter.

Verification:

mcOpenSession() returns MC_DRV_ERR_INVALID_PARAMETER, Test output is OK

Postcondition

-

Test ID:

McDrvApi_00500_mcOpenSession_BC00103_TciNullPointer

5.17.2.5 TestCase_BC00104_LenBiggerMax ( )

Description:

TLC opens a session with an invalid len parameter (len=MC_MAX_TCI_LEN+1)

Precondition

Involved components running and functioning

Focus:

mcOpenSession returns an error for a buffer exceeding length parameter.

Verification:

mcOpenSession() returns MC_DRV_ERR_INVALID_PARAMETER, Test output is OK

Postcondition

-

Test ID:

McDrvApi_00500_mcOpenSession_BC00104_LenBiggerMax

5.17.2.6 TestCase_BC00105_LenMinus1 ( )

Description:

TLC opens a session with an invalid len parameter (len=-1)

Precondition

Involved components running and functioning

Focus:

mcOpenSession returns an error for a negative length parameter.

Verification:

mcOpenSession() returns MC_DRV_ERR_INVALID_PARAMETER, Test output is OK

Postcondition

-

Test ID:

McDrvApi_00500_mcOpenSession_BC00105_LenMinus1

5.17.2.7 TestCase_BC00106_Len0 ( )

Description:

TLC opens a session with an invalid len parameter (len=0)

Precondition

Involved components running and functioning

Focus:

mcOpenSession returns an error for a zero length parameter.

Verification:

mcOpenSession() returns MC_DRV_OK, Test output is OK.

Note

The outcome is rather unexpected, is a len of 0 ok? Actually, the session is created, but the Trustlet exits immediately. Exit can be recognized with mcWaitNotification() and mcGetSessionErrorCode() However, the session has to be closed anyhow

Postcondition

-

Test ID:

McDrvApi_00500_mcOpenSession_BC00106_Len0

5.17.2.8 TestCase_BC00107_SessionhandleInvalid ( )

Description:

TLC opens a session with an invalid device id (-1)

Precondition

Involved components running and functioning

Focus:

mcOpenSession returns an error for an unknown device ID.

Verification:

mcOpenSession() returns MC_DRV_ERR_UNKNOWN_DEVICE.
Test output is OK.

Postcondition

-

Test ID:

McDrvApi_00500_mcOpenSession_BC00107_SessionhandleInvalid

5.17.2.9 TestCase_BC00108_UuidUnknown1234123412341234 ( )

Description:

TLC opens a session with an invalid UUID and that should be impossible.

Precondition

Involved components running and functioning

Focus:

mcOpenSession returns an error for an invalid uuid parameter.

Verification:

mcOpenSession() returns MC_DRV_ERR_UNKNOWN_DEVICE.
Test output is OK.

Postcondition

-

Test ID:

McDrvApi_00500_mcOpenSession_BC00108_UuidUnknown1234123412341234

5.17.2.10 TestCase_GC00109_MaximumSessions ( )

Description:

TLC opens 2 sessions, then out of memory

Precondition

Involved components running and functioning

Focus:

mcMallocWsm and mcOpenSession return errors to avoid memory exhaustion.

Verification:

Returns 2 times MC_DRV_OK, then MC_DRV_ERR_INVALID_DEVICE_FILE.
Test output is OK.

Note

This test also tests RTM ARM exception handling, at least on some boards.

Postcondition

-

Test ID:

McDrvApi_00500_mcOpenSession_GC00109_MaximumSessions

5.17.2.11 TestCase_BC00110_CloseDeviceWithOpenSession ( )

Description:

TLC tries to close a device with an open session attached to it

Precondition

Involved components running and functioning

Focus:

mcCloseDevice should return an error of a pending session.

Verification:

Returns MC_DRV_ERR_SESSION_PENDING, device and session stay open. Test output is OK.

Postcondition

-

Test ID:

McDrvApi_00500_mcOpenSession_BC00110_CloseDeviceWithOpenSession

5.17.2.12 TestCase_BC00111_LeaveSession ( )

Description:

TLC tries to open a session and leave it open

Precondition

Involved components running and functioning

Focus:

mcOpenSession should open persistent sessions.

Verification:

Test output is OK.

Postcondition

-

Test ID:

McDrvApi_00500_mcOpenSession_BC00111_LeaveSession

5.18 00600_mcCloseSession

Involved Components and their required Functionality

Functions

- TestCase_GC00100 ()
- TestCase_BC00101_SessionNullPointer ()
- TestCase_BC00102_SessionInvalidId ()
- TestCase_BC00103_SessionInvalidDevice ()
- TestCase_BC00104_CloseMultipleTimes ()

5.18.1 Detailed Description

Involved Components and their required Functionality

1. ClientLib
   (a) Can connect to daemon
   (b) Can reserve memory for connection
2. McDriverDaemon
   (a) Daemon running
   (b) Daemon socket accessible by TLC
   (c) mcOpenDevice()
3. McDriverModule
   (a) Module loaded
   (b) Module socket accessible by daemon
   (c) Module listens for connection
4. Trustlet TlIntegrationTest
   (a) Trustlet copied to Android on Test device in /data/app/ as 05010000000000000000000000000000.trst
   (b) Trustlet runnable and bug-free **Generic preconditions (setup)**
   Generic Preconditions are the same for every test case. They are executed always in front of every single test case.

- open device using the default ID by calling mcOpenDevice()
- allocate a WSM buffer (4k)
- open a session to the Trustlet

Generic post processing (teardown)

Generic post processing is the same for every test case. It is executed always after every single test case.

- close session to the Trustlet
- free allocated memory of WSM buffer
- close Device by calling mcCloseDevice()

5.18.2 Function Documentation

5.18.2.1 TestCase_GC00100 ( )

Description:

TLC closes a default device (MC_DEVICE_ID_DEFAULT) after successfully opening the device

Precondition

Involved components running and functioning

Focus:

mcCloseSession closes a session with default ID parameter.

Verification:

returns MC_DRV_OK.
Test output is OK

Postcondition

Test ID:

McDrvApi_00600_mcCloseSession_GC00100

5.18.2.2 TestCase_BC00101_SessionNullPointer ( )

Description:

TLC calls function with a NULL pointer

Precondition

Involved components running and functioning

Focus:

mcCloseSession handles the NULL paramters

Verification:

returns MC_DRV_ERR_INVALID_PARAMETER.
Test output is OK

Postcondition

Test ID:

McDrvApi_00600_mcCloseSession_BC00101_SessionNullPointer

5.18.2.3 TestCase_BC00102_SessionInvalidId ( )

Description:

TLC calls function with a malformed session id.
Ensures the unknown session ids are handled.

Precondition

Involved components running and functioning

Focus:

mcCloseSession handles a bad session id.

Verification:

returns MC_DRV_ERR_UNKNOWN_SESSION.
Test output is OK

Postcondition

Test ID:

McDrvApi_00600_mcCloseSession_BC00102_SessionInvalidId

5.18.2.4 TestCase_BC00103_SessionInvalidDevice ( )

Description:

TLC calls function with a malformed deviceId. Ensures the unknown device ids are handled.

Precondition

Involved components running and functioning

Focus:

mcCloseSession handles a bad device id.

Verification:

returns MC_DRV_ERR_UNKNOWN_DEVICE.
Test output is OK

Postcondition

Test ID:

McDrvApi_00600_mcCloseSession_BC00103_SessionInvalidDevice

5.18.2.5 TestCase_BC00104_CloseMultipleTimes ( )

Description:

TLC tries to close a device multiple times. Ensures there is no remained of previously closed session.

Precondition

Involved components running and functioning

Focus:

mcCloseSession handles an already closed session's close request.

Verification:

returns MC_DRV_ERR_UNKNOWN_SESSION.
Test output is OK

Postcondition

Test ID:

McDrvApi_00600_mcCloseSession_BC00104_CloseMultipleTimes

5.19 00700_mcNotify

Involved Components and their required Functionality

Functions

- TestCase_GC00100_notifyOnce ()
- TestCase_BC00101_SessionhandleNullPointer ()
- TestCase_BC00102_SessionhandleInvalidId ()
- TestCase_BC00103_SessionhandleInvalidDevice ()

5.19.1 Detailed Description

Involved Components and their required Functionality

1. ClientLib
   (a) Can connect to daemon
   (b) Can reserve memory for connection
2. McDriverDaemon
   (a) Daemon running
   (b) Daemon socket accessible by TLC
   (c) mcOpenDevice()
3. McDriverModule
   (a) Module loaded
   (b) Module socket accessible by daemon
   (c) Module listens for connection
4. Trustlet TIIntegrationTest
   (a) Trustlet copied to Android on Test device in /data/app/ as 05010000000000000000000000000000.trst
   (b) Trustlet runnable and bug-free

Generic preconditions (setup)

Generic Preconditions are the same for every test case. They are executed always in front of every single test case.

- open device using the default ID by calling mcOpenDevice()
- allocate a WSM buffer (4k)
- open a session to the Trustlet

Generic post processing (teardown)

Generic post processing is the same for every test case. It is executed always after every single test case.

- close session to the Trustlet
- free allocated memory of WSM buffer
- close Device by calling mcCloseDevice()

5.19.2 Function Documentation

5.19.2.1 TestCase_GC00100_notifyOnce ( )

Description:

Trustlet Connector notifies Trustlet of a new command.
Ensures the noitification has been performed successfully.

Precondition

Involved components running and functioning

Focus:

mcNotify is performed for normal parameters.

Verification:

returns MC_DRV_OK.
Test output is OK

Postcondition

Test ID:

McDrvApi_00700_mcNotify_GC00100_notifyOnce

5.19.2.2 TestCase_BC00101_SessionhandleNullPointer ( )

Description:

TLC calls function with a NULL pointer

Precondition

Involved components running and functioning

Focus:

mcNotify handles a NULL parameter.

Verification:

returns MC_DRV_ERR_INVALID_PARAMETER.
Test output is OK

Postcondition

Test ID:

McDrvApi_00700_mcNotify_BC00101_SessionhandleNullPointer

5.19.2.3 TestCase_BC00102_SessionhandleInvalidId ( )

Description:

TLC calls function with an unknown sessionId.

Precondition

Involved components running and functioning

Focus:

mcNotify handles an unknown session ID.

Verification:

returns MC_DRV_ERR_UNKNOWN_SESSION.
Test output is OK

Postcondition

Test ID:

McDrvApi_00700_mcNotify_BC00102_SessionhandleInvalidId

5.19.2.4 TestCase_BC00103_SessionhandleInvalidDevice ( )

Description:

TLC calls function with a malformed handle, bad deviceId.

Precondition

Involved components running and functioning

Focus:

mcNotify handles an unknown device ID.

Verification:

returns MC_DRV_ERR_UNKNOWN_DEVICE.
Test output is OK

Postcondition

Test ID:

McDrvApi_00700_mcNotify_BC00103_SessionhandleInvalidDevice

5.20 00800_mcWaitNotification

Involved Components and their required Functionality

Functions

- TestCase_GC00100_NotifyThenTimeoutInfinite ()
- TestCase_GC00101_NotifyThenTimeoutFinite ()
- TestCase_GC00102_NoNotifyThenTimeoutFinite ()
- TestCase_BC00103_NoNotifyThenThenNoTimeout ()
- TestCase_BC00104_SessionhandleNullPointer ()
- TestCase_BC00105_SessionhandleInvalidId ()
- TestCase_BC00106_SessionhandleInvalidDevice ()

5.20.1 Detailed Description

Involved Components and their required Functionality

1. ClientLib
   (a) Can connect to daemon
   (b) Can reserve memory for connection
2. McDriverDaemon
   (a) Daemon running
   (b) Daemon socket accessible by TLC
   (c) mcOpenDevice()
3. McDriverModule
   (a) Module loaded
   (b) Module socket accessible by daemon
   (c) Module listens for connection
4. Trustlet TlIntegrationTest
   (a) Trustlet copied to Android on Test device in /data/app/ as 05010000000000000000000000000000.trst
   (b) Trustlet runnable and bug-free

Generic preconditions (setup)

Generic Preconditions are the same for every test case. They are executed always in front of every single test case.

- open device using the default ID by calling mcOpenDevice()
- allocate a WSM buffer (4k)

- open a session to the Trustlet

Generic post processing (teardown)

Generic post processing is the same for every test case. It is executed always after every single test case.

- close session to the Trustlet
- free allocated memory of WSM buffer
- close Device by calling mcCloseDevice()

5.20.2 Function Documentation

5.20.2.1 TestCase_GC00100_NotifyThenTimeoutInfinite ( )

Description:

Trustlet Connector notifies Trustlet and waits infinitely for a notification of the Trustlet (response)

Precondition

Involved components running and functioning

Focus:

mcWaitNotification arbitrarily awaits for a notfication of an empty message.

Verification:

mcWaitNotification() returns MC_DRV_OK.
Response ID is set in message header on TCI.
No additional error code is set.
Test output is OK.

Postcondition

Test ID:

McDrvApi_00800_mcWaitNotification_GC00100_NotifyThenTimeoutInfinite

5.20.2.2 TestCase_GC00101_NotifyThenTimeoutFinite ( )

Description:

Trustlet Connector notifies Trustlet and waits 1 second for a notification of the Trustlet (response)

Precondition

Involved components running and functioning

Focus:

mcWaitNotification awaits 1 sec for a rnotification without an early timeout.

Verification:

mcWaitNotification() returns MC_DRV_OK.
Response ID is set in message header on TCI.
No additional error code is set.
Test output is OK.

Postcondition

Test ID:

McDrvApi_00800_mcWaitNotification_GC00101_NotifyThenTimeoutFinite

5.20.2.3 TestCase_GC00102_NoNotifyThenTimeoutFinite ( )

Description:

Trustlet Connector does not notify Trustlet and waits 1 second for a notification of the Trustlet (response)

Precondition

Involved components running and functioning

Focus:

Verification:

mcWaitNotification() returns MC_DRV_ERR_TIMEOUT.
Test output is OK.

Postcondition

Test ID:

McDrvApi_00800_mcWaitNotification_GC00102_NoNotifyThenTimeoutFinite

5.20.2.4 TestCase_BC00103_NoNotifyThenThenNoTimeout ( )

Description:

Trustlet Connector does not notify Trustlet and waits infinitely for a notification of the Trustlet (response)

Precondition

Involved components running and functioning

Focus:

mcWaitNotification handles a no notification case.

Verification:

mcWaitNotification() returns MC_DRV_ERR_NOTIFICATION.
Test output is OK.

Postcondition

Test ID:

McDrvApi_00800_mcWaitNotification_BC00103_NoNotifyThenThenNoTimeout

5.20.2.5 TestCase_BC00104_SessionhandleNullPointer ( )

Description:

TLC calls function with a NULL pointer

Precondition

Involved components running and functioning

Focus:

mcWaitNotification handles a NULL parameter case.

Verification:

returns MC_DRV_ERR_INVALID_PARAMETER.
Test output is OK

Postcondition

Test ID:

McDrvApi_00800_mcWaitNotification_BC00104_SessionhandleNullPointer

5.20.2.6 TestCase_BC00105_SessionhandleInvalidId ( )

Description:

TLC calls function with a malformed handle, bad sessionId.

Precondition

Involved components running and functioning

Focus:

mcWaitNotification handles an invalid session ID case.

Verification:

returns MC_DRV_ERR_UNKNOWN_SESSION.
Test output is OK

Postcondition

Test ID:

McDrvApi_00800_mcWaitNotification_BC00105_SessionhandleInvalidId

5.20.2.7 TestCase_BC00106_SessionhandleInvalidDevice ( )

Description:

TLC calls function with a malformed handle, bad deviceId.

Precondition

Involved components running and functioning

Focus:

mcWaitNotification handles an invalid device ID case.

Verification:

returns MC_DRV_ERR_UNKNOWN_DEVICE.
Test output is OK

Postcondition

Test ID:

McDrvApi_00800_mcWaitNotification_BC00106_SessionhandleInvalidDevice

5.21 00900_mcGetSessionErrorCode

Involved Components and their required Functionality

Functions

- TestCase_GC00050_NoNotifications ()
- TestCase_GC00100_MultipleNotifications ()
- TestCase_GC00101_TrustletExits ()
- TestCase_GC00103_TrustletCrashes ()
- TestCase_BC00104_SessionhandleNullPointer ()
- TestCase_BC00105_LastErrNullPointer ()
- TestCase_BC00106_SessionhandleInvalidId ()
- TestCase_BC00107_SessionhandleInvalidDevice ()

5.21.1 Detailed Description

Involved Components and their required Functionality

1. ClientLib
   (a) Can connect to daemon
   (b) Can reserve memory for connection
2. McDriverDaemon
   (a) Daemon running
   (b) Daemon socket accessible by TLC
   (c) mcOpenDevice()
3. McDriverModule
   (a) Module loaded
   (b) Module socket accessible by daemon
   (c) Module listens for connection
4. Trustlet TlIntegrationTest
   (a) Trustlet copied to Android on Test device in /data/app/ as 05010000000000000000000000000000.trst
   (b) Trustlet runnable and bug-free

Generic preconditions (setup)

Generic Preconditions are the same for every test case. They are executed always in front of every single test case.

- open device using the default ID by calling mcOpenDevice()

- allocate a WSM buffer (4k)
- open a session to the Trustlet

Generic post processing (teardown)

Generic post processing is the same for every test case. It is executed always after every single test case.

- close session to the Trustlet
- free allocated memory of WSM buffer
- close Device by calling mcCloseDevice()

5.21.2 Function Documentation

5.21.2.1 TestCase_GC00050_NoNotifications ( )

Description:

Ensures that no session error is returned if there is none.

Precondition

Involved components running and functioning

Focus:

mcGetSessionErrorCode() returns MC_DRV_OK if there is no error.

Verification:

mcGetSessionErrorCode() returns MC_DRV_OK.
No additional error code is set.
Test output is OK.

Postcondition

Test ID:

McDrvApi_00900_mcGetSessionErrorCode_GC00050_NoNotifications

5.21.2.2 TestCase_GC00100_MultipleNotifications ( )

Description:

Ensures that trustlet Connector handles multiple notification of test trustlet successfully.

Precondition

Involved components running and functioning

Focus:

mcWaitNotification handles 3 notifications of trustlet.

Verification:

mcWaitNotification() returns MC_DRV_OK.
Response ID is set in message header on TCI.
No additional error code is set.
Test output is OK.

Postcondition

Test ID:

McDrvApi_00900_mcGetSessionErrorCode_GC00100_MultipleNotifications

5.21.2.3 TestCase_GC00101_TrustletExits ( )

Description:

Trustlet Connector handles the "last" notification of the Trustlet after the Trustlet terminates itself with a wrong exit code.

Precondition

Involved components running and functioning

Focus:

mcGetSessionErrorCode is ERR_INVALID_EXIT_CODE as the last error.

Verification:

mcWaitNotification() returns MC_DRV_INFO_NOTIFICATION.
The additional error code is set to -1: invalid exit code.
Test output is OK.

Postcondition

Test ID:

McDrvApi_00900_mcGetSessionErrorCode_GC00101_TrustletExits

5.21.2.4 TestCase_GC00103_TrustletCrashes ( )

Description:

Trustlet Connector handles the notification of the TL after the TL causes a kernel exception.

Precondition

Involved components running and functioning

Focus:

mcGetSessionErrorCode() returns "ERR_INVALID_OPERATION"

Verification:

mcWaitNotification() returns MC_DRV_INFO_NOTIFICATION.
Test output is OK.

Postcondition

Test ID:

McDrvApi_00900_mcGetSessionErrorCode_GC00103_TrustletCrashes

5.21.2.5 TestCase_BC00104_SessionhandleNullPointer ( )

Description:

TLC calls function with a NULL pointer session

Precondition

Involved components running and functioning

Focus:

mcGetSessionErrorCode handles a null session.

Verification:

returns MC_DRV_ERR_INVALID_PARAMETER.
Test output is OK

Postcondition

Test ID:

McDrvApi_00900_mcGetSessionErrorCode_BC00104_SessionhandleNullPointer

5.21.2.6 TestCase_BC00105_LastErrNullPointer ( )

Description:

TLC calls function with a NULL pointer lastErr

Precondition

Involved components running and functioning

Focus:

mcGetSessionErrorCode handles a null last error pointer.

Verification:

returns MC_DRV_ERR_INVALID_PARAMETER.
Test output is OK

Postcondition

Test ID:

McDrvApi_00900_mcGetSessionErrorCode_BC00105_LastErrNullPointer

5.21.2.7 TestCase_BC00106_SessionhandleInvalidId ( )

Description:

TLC calls function with a malformed handle, bad sessionId.

Precondition

Involved components running and functioning

Focus:

mcGetSessionErrorCode handles an unknown session ID.

Verification:

returns MC_DRV_ERR_UNKNOWN_SESSION.
Test output is OK

Postcondition

Test ID:

McDrvApi_00900_mcGetSessionErrorCode_BC00106_SessionhandleInvalidId

5.21.2.8 TestCase_BC00107_SessionhandleInvalidDevice ( )

Description:

TLC calls function with a malformed handle, bad deviceId.

Precondition

Involved components running and functioning

Focus:

mcGetSessionErrorCode handles an unknown device ID.

Verification:

returns MC_DRV_ERR_UNKNOWN_DEVICE.
Test output is OK

Postcondition

Test ID:

McDrvApi_00900_mcGetSessionErrorCode_BC00107_SessionhandleInvalidDevice

5.22 01000_mcMap

Involved Components and their required Functionality

Functions

- TestCase_GC00100_Map4k ()
- TestCase_BC00101_SessionhandleNullPointer ()
- TestCase_BC00102_MapinfoNullPointer ()
- TestCase_BC00103_SessionhandleInvalidId ()
- TestCase_BC00104_SessionhandleInvalidDevice ()
- TestCase_BC00105_MultipleMappingsOfSameBuffer ()
- TestCase_BC00106_MapAReadonlyBuffer ()

5.22.1 Detailed Description

Involved Components and their required Functionality

1. ClientLib
   (a) Can connect to daemon
   (b) Can reserve memory for connection

2. McDriverDaemon
   (a) Daemon running
   (b) Daemon socket accessible by TLC
   (c) mcOpenDevice()
3. McDriverModule
   (a) Module loaded
   (b) Module socket accessible by daemon
   (c) Module listens for connection
4. Trustlet TIIntegrationTest
   (a) Trustlet copied to Android on Test device in /data/app/ as 05010000000000000000000000000000.trst
   (b) Trustlet runnable and bug-free

Generic preconditions (setup)

Generic Preconditions are the same for every test case. They are executed always in front of every single test case.

- open device using the default ID by calling mcOpenDevice()
- allocate a WSM buffer (4k)
- open a session to the Trustlet

Generic post processing (teardown)

Generic post processing is the same for every test case. It is executed always after every single test case.

- close session to the Trustlet
- free allocated memory of WSM buffer
- close Device by calling mcCloseDevice()

5.22.2 Function Documentation

5.22.2.1 TestCase_GC00100_Map4k ( )

Description:

TLC maps additional Bulk memory to Trustlet

Precondition

Involved components running and functioning

Focus:

mcMap does not map additional bulk memory.

Verification:

returns MC_DRV_OK.
returned Buffer length is equal to requested length.
Test output is OK.

Postcondition

Test ID:

McDrvApi_01000_mcMap_GC00100_Map4k

5.22.2.2 TestCase_BC00101_SessionhandleNullPointer ( )

Description:

TLC calls function with a NULL pointer session

Precondition

Involved components running and functioning

Focus:

mcMap handles a null session pointer.

Verification:

returns MC_DRV_ERR_INVALID_PARAMETER.
Test output is OK

Postcondition

Test ID:

McDrvApi_01000_mcMap_BC00101_SessionhandleNullPointer

5.22.2.3 TestCase_BC00102_MapinfoNullPointer ( )

Description:

TLC calls function with a NULL pointer mapInfo

Precondition

Involved components running and functioning

Focus:

mcMap handles a null map info pointer.

Verification:

returns MC_DRV_ERR_INVALID_PARAMETER.
Test output is OK

Postcondition

Test ID:

McDrvApi_01000_mcMap_BC00102_MapinfoNullPointer

5.22.2.4 TestCase_BC00103_SessionhandleInvalidId ( )

Description:

TLC calls function with a malformed handle, bad sessionId.

Precondition

Involved components running and functioning

Focus:

mcMap handles an invalid session ID.

Verification:

returns MC_DRV_ERR_UNKNOWN_SESSION.
Test output is OK

Postcondition

Test ID:

McDrvApi_01000_mcMap_BC00103_SessionhandleInvalidId

5.22.2.5 TestCase_BC00104_SessionhandleInvalidDevice ( )

Description:

TLC calls function with a malformed handle, bad deviceId.

Precondition

Involved components running and functioning

Focus:

mcMap handles an unknown device id.

Verification:

returns MC_DRV_ERR_UNKNOWN_DEVICE.
Test output is OK

Postcondition

Test ID:

McDrvApi_01000_mcMap_BC00104_SessionhandleInvalidDevice

5.22.2.6 TestCase_BC00105_MultipleMappingsOfSameBuffer ( )

Description:

TLC maps the same additional bulk memory to one Trustlet twice

Precondition

Involved components running and functioning

Focus:

mcMap handles a multiple memory mapping on same Trustlet.

Verification:

returns MC_DRV_ERR_BULK_MAPPING.
The bulk buffer cannot be mapped to the same address.
Test output is OK.

Postcondition

Test ID:

McDrvApi_01000_mcMap_BC00105_MultipleMappingsOfSameBuffer

5.22.2.7 TestCase_BC00106_MapAReadonlyBuffer ( )

Description:

TLC tries to map read-only data to Trustlet

Precondition

Involved components running and functioning

Focus:

mcMap handles a readonly memory area mapping.

Verification:

returns MC_DRV_ERR_BULK_MAPPING.
The bulk buffer cannot be mapped because it points to readonly data.
Test output is OK.

Postcondition

Test ID:

McDrvApi_01000_mcMap_BC00106_MapAReadonlyBuffer

5.23 01100_mcUnmap

Involved Components and their required Functionality

Functions

- TestCase_GC00100_Unmap4k ()
- TestCase_BC00101_SessionhandleNullPointer ()
- TestCase_BC00102_BufNullPointer ()
- TestCase_BC00103_MapinfoNullPointer ()
- TestCase_BC00104_SessionhandleInvalidId ()
- TestCase_BC00105_SessionhandleInvalidDevice ()
- TestCase_BC00106_BufferPointerNotManagedByLib ()
- TestCase_BC00107_MapinfoAddress0 ()
- TestCase_BC00108_MapinfoAddressMinus1 ()
- TestCase_BC00110_UnmapAnAlreadyUnmappedBulkBuffer ()
- TestCase_GC00111_UnmapDoesntClear ()

5.23.1 Detailed Description

Involved Components and their required Functionality

1. ClientLib
   (a) Can connect to daemon
   (b) Can reserve memory for connection
2. McDriverDaemon
   (a) Daemon running

(b) Daemon socket accessible by TLC

(c) mcOpenDevice()

3. McDriverModule

(a) Module loaded

(b) Module socket accessible by daemon

(c) Module listens for connection

4. Trustlet TlIntegrationTest

(a) Trustlet copied to Android on Test device in /data/app/ as 05010000000000000000000000000000.trst

(b) Trustlet runnable and bug-free

5.23.2 Function Documentation

5.23.2.1 TestCase_GC00100_Unmap4k ( )

Description:

The previous mapped bulk buffer is unmapped from Trustlet

Precondition

Involved components running and functioning

Focus:

mcUnMap unmaps successfully with the expected inputs.

Verification:

returns MC_DRV_OK.

Postcondition

Test output is OK.

Test ID:

McDrvApi_01100_mcUnmap_GC00100_Unmap4k

5.23.2.2 TestCase_BC00101_SessionhandleNullPointer ( )

Description:

TLC calls function with a NULL pointer session

Precondition

Involved components running and functioning

Focus:

mcUnMap handles a null session pointer.

Verification:

returns MC_DRV_ERR_INVALID_PARAMETER.
Test output is OK

Postcondition

Test ID:

McDrvApi_01100_mcUnmap_BC00101_SessionhandleNullPointer

5.23.2.3 TestCase_BC00102_BufNullPointer ( )

Description:

TLC calls function with a NULL pointer bulkBuffer

Precondition

Involved components running and functioning

Focus:

mcUnMap handles a null bulk data pointer.

Verification:

returns MC_DRV_ERR_INVALID_PARAMETER.
Test output is OK

Postcondition

Test ID:

McDrvApi_01100_mcUnmap_BC00102_BufNullPointer

5.23.2.4 TestCase_BC00103_MapinfoNullPointer ( )

Description:

TLC calls function with a NULL pointer mapInfo

Precondition

Involved components running and functioning

Focus:

mcUnMap handles a null buffer info pointer.

Verification:

returns MC_DRV_ERR_INVALID_PARAMETER.
Test output is OK

Postcondition

Test ID:

McDrvApi_01100_mcUnmap_BC00103_MapinfoNullPointer

5.23.2.5 TestCase_BC00104_SessionhandleInvalidId ( )

Description:

TLC calls function with a malformed handle, bad sessionId.

Precondition

Involved components running and functioning

Focus:

mcUnMap handles a unknown session ID.

Verification:

returns MC_DRV_ERR_UNKNOWN_SESSION.
Test output is OK

Postcondition

Test ID:

McDrvApi_01100_mcUnmap_BC00104_SessionhandleInvalidId

5.23.2.6 TestCase_BC00105_SessionhandleInvalidDevice ( )

Description:

TLC calls function with a malformed handle, bad deviceId.

Precondition

Involved components running and functioning

Focus:

mcUnMap handles an invalid device ID.

Verification:

returns MC_DRV_ERR_UNKNOWN_DEVICE.
Test output is OK

Postcondition

Test ID:

McDrvApi_01100_mcUnmap_BC00105_SessionhandleInvalidDevice

5.23.2.7 TestCase_BC00106_BufferPointerNotManagedByLib ( )

Description:

TLC calls function with a bad bulkBuf.

Precondition

Involved components running and functioning

Focus:

mcUnMap handles an oversized bulk buffer.

Verification:

returns MC_DRV_ERR_BULK_UNMAPPING.
Test output is OK

Postcondition

Test ID:

McDrvApi_01100_mcUnmap_BC00106_BufferPointerNotManagedByLib

5.23.2.8 TestCase_BC00107_MapinfoAddress0 ( )

Description:

TLC calls function with a malformed mapInfo

Precondition

Involved components running and functioning

Focus:

mcUnMap handles a null virtual addressed map info.

Verification:

returns MC_DRV_ERR_BULK_UNMAPPING.
Test output is OK.

Note

This test also tests RTM ARM exception handling.

Postcondition

Test ID:

McDrvApi_01100_mcUnmap_BC00107_MapinfoAddress0

5.23.2.9 TestCase_BC00108_MapinfoAddressMinus1 ( )

Description:

TLC calls function with a malformed mapInfo

Precondition

Involved components running and functioning

Focus:

mcUnMap handles a bad virtual addressed map info.

Verification:

returns MC_DRV_ERR_BULK_UNMAPPING.
Test output is OK.

Note

This test also tests RTM ARM exception handling.

Postcondition

Test ID:

McDrvApi_01100_mcUnmap_BC00108_MapinfoAddressMinus1

5.23.2.10 TestCase_BC00110_UnmapAnAlreadyUnmappedBulkBuffer ( )

Description:

TLC unmaps the same bulk buffer from Trustlet twice

Precondition

Involved components running and functioning

Focus:

mcUnMap handles a multiple unmapping on the same session.

Verification:

returns MC_DRV_ERR_BULK_UNMAPPING.

Postcondition

Test ID:

McDrvApi_01100_mcUnmap_BC00110_UnmapAnAlreadyUnmappedBulkBuffer

5.23.2.11 TestCase_GC00111_UnmapDoesntClear ( )

Description:

TLC writes something in buffer and then maps and unmaps the buffer to Trustlet

Precondition

Involved components running and functioning

Focus:

mcMap and mcUnMap handle an already written bulk buffer.

Verification:

Unmapping does not clear the buffer
Test output is OK.

Postcondition

Test ID:

McDrvApi_01100_mcUnmap_GC00111_UnmapDoesntClear

5.24 01200_mcGetRndSeed

Involved Components and their required Functionality

Functions

- TestCase_GC01200_GetRngSeedByDriver ()

5.24.1 Detailed Description

Involved Components and their required Functionality

1. ClientLib
   (a) Can connect to daemon
   (b) Can reserve memory for connection
2. McDriverDaemon
   (a) Daemon running
   (b) Daemon socket accessible by TLC
   (c) mcOpenDevice()
3. McDriverModule
   (a) Module loaded
   (b) Module socket accessible by daemon
   (c) Module listens for connection
4. Trustlet TlIntegrationTest
   (a) Trustlet copied to Android on Test device in /data/app/ as 05010000000000000000000000000000.trst
   (b) Trustlet runnable and bug-free

5.24.2 Function Documentation

5.24.2.1 TestCase_GC01200_GetRngSeedByDriver ( )

Description:

TLC opens a session,

Precondition

Focus:

mcNotify and mcWaitNotification return MC_DRV_OK

Verification:

mapped area includes non-zero values after RNG seed read operation successfully performed.

Postcondition

Test ID:

McDrvApi_01200_mcGetRngSeed_GC01200_GetRngSeedByDriver

5.25 Runtime

Test of the runtime behavior of SWd components.

Modules

- 00100_TrustletCommands

  Involved Components and their required Functionality

- 00300_Dead_Trustlet

  Involved Components and their required Functionality

- 00301_Trustlet_Endless_Loop

  Involved Components and their required Functionality

- 00400_Communication

  Parallel Notifications and Mappings.

- 00500_DynamicDriver

  Involved Components and their required Functionality

- 00502_DynamicDriver3

  Involved Components and their required Functionality

- 00503_DynamicDriver4

  Involved Components and their required Functionality

- 00600_CpuCoreAvailability

  Involved Components and their required Functionality

- Concurrency Tests
- Trustlet Communication Tests

5.25.1 Detailed Description

Test of the runtime behavior of SWd components.

5.26 00100_TrustletCommands

Involved Components and their required Functionality

Functions

- TestCase_BC00100_TrustletCommandHasRespIdSet ()
- TestCase_BC00101_UnknownTestCommand ()
- TestCase_GC00102_DoNothing ()

5.26.1 Detailed Description

Involved Components and their required Functionality

1. ClientLib
   (a) Can connect to daemon
   (b) Can reserve memory for connection
2. McDriverDaemon
   (a) Daemon running
   (b) Daemon socket accessible by TLC
   (c) mcOpenDevice()
3. McDriverModule
   (a) Module loaded
   (b) Module socket accessible by daemon
   (c) Module listens for connection
4. Trustlet TlIntegrationTest
   (a) Trustlet copied to Android on Test device in /data/app/ as 05010000000000000000000000000000.trst

5.26.2 Function Documentation

5.26.2.1 TestCase_BC00100_TrustletCommandHasRespIdSet ( )

Description:

Trustlet Connector notifies Trustlet and waits infinitely for a notification of the Trustlet (response) TLC uses a bad command, with RSP_ID set. Trustlet just answers back, no error message

Precondition

Session has been opened by setup()

Focus:

Use mcNotify() to send a valid command to the trustlet
Wait for notification from Trustlet

Verification:

mcWaitNotification() returns MC_DRV_OK.
No additional error code is set.

Postcondition

Session is closed by teardown()

Test ID:

Runtime_00100_TrustletCommands_BC00100_TrustletCommandHasRespIdSet

5.26.2.2 TestCase_BC00101_UnknownTestCommand ( )

Description:

Trustlet Connector notifies Trustlet and waits infinitely for a notification of the Trustlet (response) TLC uses a bad command, with RSP_ID set. Trustlet just answers back, no error message

Precondition

Session has been opened by setup()

Focus:

Use mcNotify() to send an unknown command to the trustlet
Wait for notification from Trustlet

Verification:

mcWaitNotification() returns MC_DRV_OK.
check that tci->responseHeader.returncode == RET_ERR_UNKNOWN_CMD
No additional error code is set.

Postcondition

Session is closed by teardown()

Test ID:

Runtime_00100_TrustletCommands_BC00101_UnknownTestCommand

5.26.2.3 TestCase_GC00102_DoNothing ( )

Description:

Trustlet does nothing (Immediate open/close session). Setup() opens the session. TestCase does nothing and Teardown() closes the session.

Precondition

Session has been opened by setup()

Focus:

-

Verification:

-

Postcondition

Session is closed by teardown()

Test ID:

Runtime_00100_TrustletCommands_GC00102_DoNothing

5.27 00300_Dead_Trustlet

Involved Components and their required Functionality

Functions

- TEST_SCENARIO (00300_Dead_Trustlet)
- TestCase_BC00100_NotifyDeadTrustletWaitInfinite ()
- TestCase_BC00101_MapMemoryToDeadTrustlet ()

5.27.1 Detailed Description

Involved Components and their required Functionality

1. ClientLib
   (a) Can connect to daemon
   (b) Can reserve memory for connection
2. McDriverDaemon
   (a) Daemon running

(b) Daemon socket accessible by TLC

(c) mcOpenDevice()

3. McDriverModule

(a) Module loaded

(b) Module socket accessible by daemon

(c) Module listens for connection

4. Trustlet TlIntegrationTest

(a) Trustlet copied to Android on Test device in /data/app/ as 05010000000000000000000000000000.trst

(b) Trustlet runnable and bug-free

5.27.2 Function Documentation

5.27.2.1 TEST_SCENARIO (00300_Dead_Trustlet)

Setup: TLC handles the "last" notification of the Trustlet after the Trustlet terminates itself with a wrong exit code.

5.27.2.2 TestCase_BC00100_NotifyDeadTrustletWaitInfinite ( )

Description:

Afterwards TLC tries to notify the Trustlet Then it waits for the Trustlet

Precondition

Involved components running and functioning

Verification:

mcWaitNotification() returns MC_DRV_INFO_NOTIFICATION.
The additional error code is set to -1: invalid exit code.
mcNotify returns no error mcWaitNotification returns an error Test output is OK.

Test ID:

Runtime_00300_Dead_Trustlet_BC00100_NotifyDeadTrustletWaitInfinite

5.27.2.3 TestCase_BC00101_MapMemoryToDeadTrustlet ( )

Description:

Afterwards TLC tries to map bulk memory to the Trustlet

See also Test10_mcMap_BC_pagefault_on_wrong_address for a Test of mcUnmap() on a dead trustlet.

Precondition

Involved components running and functioning

Verification:

mcMap() returns MC_DRV_ERR_DAEMON_UNREACHABLE.

Note

This test also tests RTM ARM exception handling. Test output is OK.

Test ID:

Runtime_00300_Dead_Trustlet_BC00101_MapMemoryToDeadTrustlet

5.28 00301_Trustlet_Endless_Loop

Involved Components and their required Functionality

Functions

- TestCase_BC00100_TrustletInEndlessLoop ()

5.28.1 Detailed Description

Involved Components and their required Functionality

1. ClientLib
   (a) Can connect to daemon
   (b) Can reserve memory for connection
2. McDriverDaemon
   (a) Daemon running
   (b) Daemon socket accessible by TLC
   (c) mcOpenDevice()
3. McDriverModule
   (a) Module loaded
   (b) Module socket accessible by daemon
   (c) Module listens for connection

4. Trustlet TlIntegrationTest
   (a) Trustlet copied to Android on Test device in /data/app/ as 05010000000000000000000000000000.trst
   (b) Trustlet runnable and bug-free

5.28.2 Function Documentation

5.28.2.1 TestCase_BC00100_TrustletInEndlessLoop ( )

Description:

TLC tells Trustlet to go into infinite loop. Then TLC closes Session.

Precondition

Involved components running and functioning

Verification:

Session can be closed. Test output is OK.

Test ID:

Runtime_00301_Trustlet_Endless_Loop_BC00100_TrustletInEndlessLoop

5.29 00400_Communication

Parallel Notifications and Mappings.

Functions

- TestCase_GC00100_NoNotifyThenWait ()
- TestCase_GC00101_NotifyT2ThenWait ()
- TestCase_GC00102_NotifyT1ThenWait ()
- TestCase_GC00103_NotifyT1T2ThenWait ()
- TestCase_GC00104_NotifyT2T1ThenWait ()
- TestCase_GC00105_SharedMemoryT1WritesT2Reads ()
- TestCase_GC00106_UnmapSharedMemoryFromT1ThenCheckT2CanStillRead ()

5.29.1 Detailed Description

Parallel Notifications and Mappings. **Involved Components and their required Functionality**

1. ClientLib
   (a) Can connect to daemon

(b) Can reserve memory for connection

2. McDriverDaemon

   (a) Daemon running

   (b) Daemon socket accessible by TLC

   (c) mcOpenDevice()

3. McDriverModule

   (a) Module loaded

   (b) Module socket accessible by daemon

   (c) Module listens for connection

4. Trustlet TlIntegrationTest

   (a) Trustlet copied to Android on Test device in /data/app/ as 05010000000000000000000000000000.trst

   (b) Trustlet runnable and bug-free

5.29.2 Function Documentation

5.29.2.1 TestCase_GC00100_NoNotifyThenWait ( )

Description:

TLC sends and receives notifications from two Trustlets.
In all cases it waits for both trustlet.
But it notifies only none, one, or both

Precondition

Involved components running and functioning

Verification:

mcWaitNotification() returns MC_DRV_OK or MC_DRV_ERR_TIMEOUT.
Response ID is set in message header on TCI.
Exit code is OK.
Test output is OK.

Note

This test also tests RTM ARM exception handling.

Test ID:

Runtime_00400_Communication_GC00100_NoNotifyThenWait

5.29.2.2 TestCase_GC00101_NotifyT2ThenWait ( )

Test ID:

Runtime_00400_Communication_GC00101_NotifyT2ThenWait

5.29.2.3 TestCase_GC00102_NotifyT1ThenWait ( )

Test ID:

Runtime_00400_Communication_GC00102_NotifyT1ThenWait

5.29.2.4 TestCase_GC00103_NotifyT1T2ThenWait ( )

Test ID:

Runtime_00400_Communication_GC00103_NotifyT1T2ThenWait

5.29.2.5 TestCase_GC00104_NotifyT2T1ThenWait ( )

Test ID:

Runtime_00400_Communication_GC00104_NotifyT2T1ThenWait

5.29.2.6 TestCase_GC00105_SharedMemoryT1WritesT2Reads ( )

Description:

TLC maps buffer to two Trustlets
Trustlet 1 writes in buffer.
Trustlet 2 reads the buffer.

Precondition

Involved components running and functioning

Verification:

Trustlet 2 reads what Trustlet 1 wrote.
Test output is OK.

Test ID:

Runtime_00400_Communication_GC00105_SharedMemoryT1WritesT2Reads

5.29.2.7 TestCase_GC00106_UnmapSharedMemoryFromT1ThenCheckT2CanStillRead ( )

Description:

TLC writes something in buffer and then maps it to two Trustlets.
Then TLC unmaps the buffer from one Trustlet.
The other Trustlet can still read the buffer.

Precondition

Involved components running and functioning

Verification:

Unmapping does not unmap the buffer everywhere.
Test output is OK.

Test ID:

Runtime_00400_Communication_GC00106_UnmapSharedMemoryFromT1ThenCheckT2CanStillRead

5.30 00500_DynamicDriver

Involved Components and their required Functionality

Functions

- TestCase_GC00100_DriverAllocOk ()
- TestCase_BC00101_DriverAllocTooMuch ()
- TestCase_GC00102_DoNothing ()

5.30.1 Detailed Description

Involved Components and their required Functionality

1. ClientLib
   (a) Can connect to daemon
   (b) Can reserve memory for connection
2. McDriverDaemon
   (a) Daemon running
   (b) Daemon socket accessible by TLC
   (c) mcOpenDevice()
3. DrIntegrationTest Driver
   (a) Module loaded

(b) Module socket accessible by daemon

(c) Module listens for connection

4. Trustlet TlTestIntegrationTest

(a) Trustlet copied to Android on Test device in /data/app/ as 05010000000000000000000000000000.trst

(b) Trustlet runnable and bug-free

5.30.2 Function Documentation

5.30.2.1 TestCase_GC00100_DriverAllocOk ( )

Description:

Trustlet Connector notifies Trustlet and allocate 1 resource Trustlet just answers with accept message, no error message

Precondition

Involved components running and functioning

Verification:

mcWaitNotification() returns MC_DRV_OK.

Verification:

returnCode == RET_OK No additional error code is set.
Test output is OK.

Test ID:

Runtime_00500_DynamicDriver_GC00100_DriverAllocOk

5.30.2.2 TestCase_BC00101_DriverAllocTooMuch ( )

Description:

Trustlet Connector notifies Trustlet and allocate 2 resource Trustlet answers first with accept message, no error message Trustlet answers first with not accept message, error message

Precondition

Involved components running and functioning

Verification:

mcWaitNotification() returns MC_DRV_OK.

Verification:

first returnCode == RET_OK No additional error code is set.
Test output is OK.

Test ID:

Runtime_00500_DynamicDriver_BC00101_DriverAllocTooMuch

5.30.2.3 TestCase_GC00102_DoNothing ( )

Description:

Testcase without doing anything Open/CloseSession without any duration between

Precondition

Involved components running and functioning

Verification:

mcWaitNotification() returns MC_DRV_OK.

Verification:

first returnCode == RET_OK No additional error code is set.
Test output is OK.

Test ID:

Runtime_00500_DynamicDriver_GC00102_DoNothing

5.31 00502_DynamicDriver3

Involved Components and their required Functionality

Functions

- TEST_SCENARIO (00502_DynamicDriver3)
- TestCase_GC00100_DriverMemoryMapping ()

5.31.1 Detailed Description

Involved Components and their required Functionality

1. ClientLib
   (a) Can connect to daemon

(b) Can reserve memory for connection

2. McDriverDaemon
   (a) Daemon running
   (b) Daemon socket accessible by TLC
   (c) mcOpenDevice()
3. DrIntegrationTest Driver
   (a) Module loaded
   (b) Module socket accessible by daemon
   (c) Module listens for connection
4. Trustlet TlTestIntegrationTest
   (a) Trustlet copied to Android on Test device in /data/app/ as 05010000000000000000000000000000.trst
   (b) Trustlet runnable and bug-free

5.31.2 Function Documentation

5.31.2.1 TEST_SCENARIO (00502_DynamicDriver3)

!!THP faked driver as trustlet

5.31.2.2 TestCase_GC00100_DriverMemoryMapping ( )

Description:

Trustlet Connector opens two Trustlets

Precondition

Involved components running and functioning Testflow.

Test ID:

Runtime_00502_DynamicDriver3_GC00100_DriverMemoryMapping

5.32 00503_DynamicDriver4

Involved Components and their required Functionality

Functions

- TestCase_GC00100_MultipleDriverLoading_WithoutCall ()
- TestCase_GC00101_MultipleDriverLoading_WithCall ()
- TestCase_GC00102_MultipleDriverLoading_WithCall_2Trustlets ()

5.32.1 Detailed Description

Involved Components and their required Functionality

1. ClientLib
   (a) Can connect to daemon
   (b) Can reserve memory for connection
2. McDriverDaemon
   (a) Daemon running
   (b) Daemon socket accessible by TLC
   (c) mcOpenDevice()
3. DrIntegrationTest Driver
   (a) Module loaded
   (b) Module socket accessible by daemon
   (c) Module listens for connection
4. Trustlet TlTestIntegrationTest
   (a) Trustlet copied to Android on Test device in /data/app/ as 05010000000000000000000000000000.trst
   (b) Trustlet runnable and bug-free

5.32.2 Function Documentation

5.32.2.1 TestCase_GC00100_MultipleDriverLoading_WithoutCall ( )

Description:

Trustlet Connector reopens multiple time trustlet+driver with Ping to trustlet only

Precondition

Involved components running and functioning Testflow.

Test ID:

Runtime_00503_DynamicDriver4_GC00100_MultipleDriverLoading_WithoutCall

!!THP faked driver as trustlet

5.32.2.2 TestCase_GC00101_MultipleDriverLoading_WithCall ( )

Description:

Trustlet Connector reopens multible time trustlet+driver with ping trustlet->driver

Precondition

Involved components running and functioning Testflow.

Test ID:

Runtime_00503_DynamicDriver4_GC00101_MultipleDriverLoading_WithCall

!!THP faked driver as trustlet

5.32.2.3 TestCase_GC00102_MultipleDriverLoading_WithCall_2Trustlets ( )

Description:

Trustlet Connector reopens multible time trustlet+driver with ping trustlet->driver with 2 trustlets

Precondition

Involved components running and functioning Testflow.

Test ID:

Runtime_00503_DynamicDriver4_GC00102_MultipleDriverLoading_WithCall_2Trustlets

!!THP faked driver as trustlet

5.33 00600_CpuCoreAvailability

Involved Components and their required Functionality

Functions

- TestCase_GC00100_PlugOut_and_PlugIn_Cpu1 ()

5.33.1 Detailed Description

Involved Components and their required Functionality

1. ClientLib
   (a) Can connect to daemon
   (b) Can reserve memory for connection
2. McDriverDaemon
   (a) Daemon running
   (b) Daemon socket accessible by TLC
   (c) mcOpenDevice()

5.33.2 Function Documentation

5.33.2.1 TestCase_GC00100_PlugOut_and_PlugIn_Cpu1 ( )

Description:

Mobicore caused other secondary cpu boots to fail when the process runs on Cpu1 this test has been developed to make sure if a process over Cpu1 shuts down another secondary
cpu can hotplug the core back up again. Important note: Since Power Management (PM) aggressively takes the un-required cores down
this test results as passed when PM is enabled, yet the test is not run actually.

Precondition

Running linux kernel

Focus:

fread returns 1 for Cpu2 from /sys/devices/system/cpu/cpu2/online

Verification:

read ingredients of /sys/devices/system/cpu/cpu2/online after plugging out and plugging in.

Postcondition

Test ID:

Runtime_00600_CpuCoreAvailability_GC00100_PlugOut_and_PlugIn_Cpu1

5.34 Concurrency Tests

5.35 Trustlet Communication Tests

5.36 Trustlet API (TlApi)

These tests contain good cases and bad cases for the API functions.

Modules

- Inter-world communication functions [COM].
- Security support functions [SEC].
- MobiCore system functions [SYS].

5.36.1 Detailed Description

These tests contain good cases and bad cases for the API functions. Some Trustlet API functions are also tested implicitly by other test subjects like *Trustlet Runtime* and *Performance*.

Introduction

Tests checking the basic functionality of the Trustlet interface.

5.37 MCD_MCDIMPL_DAEMON

Files

- file MobiCoreDriverCmd.h

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1.

Defines

- #define MC_DEVICE_ID_DEFAULT 0

  The default device ID.

5.38 TLAPI_SEC

Files

- file TlApiCrypto.c

  TlApi security functions.

5.39 TlApi

Data Structures

- struct tlApiCrAbort_t

  Marshaled function parameters.

- struct marshalingParam_t

  Marshaled union.

Files

- file TlApiMarshal.h

  Marshaling types and declarations.

Defines

- #define MAX_MAR_LIST_LENGTH 8

  Maximum number of parameter .

Typedefs

- typedef void * void_ptr

  a pointer to anything.

- typedef void_ptr addr_t

  an address, can be physical or virtual

- typedef addr_t buffer_t

  a data buffer

Enumerations

- enum kpdFuncID_t {
  FID_KPD_GRAB_KEYPAD = 0,
  FID_KPD_GET_C,
  FID_KPD_GET_MULTIPLE_C,
  FID_KPD_RELEASE_KEYPAD }

  Each function must have its own ID.

- enum cryptoFuncID_t {
  FID_CRYPTO_MD = 0,
  FID_CRYPTO_MD_INIT = 1,
  FID_CRYPTO_MD_INIT_WITH_DATA = 2,
  FID_CRYPTO_MD_UPDATE = 3,
  FID_CRYPTO_MD_DOFINAL = 4,
  FID_CRYPTO_SIG = 5,
  FID_CRYPTO_SIG_INIT = 6,
  FID_CRYPTO_SIG_INIT_WITH_DATA = 7,

FID_CRYPTO_SIG_UPDATE = 8,
FID_CRYPTO_SIG_SIGN = 9,
FID_CRYPTO_SIG_VERIFY = 10,
FID_CRYPTO_RNG = 11,
FID_CRYPTO_RNG_GENERATE_DATA = 12,
FID_CRYPTO_CIPHER = 13,
FID_CRYPTO_CIPHER_INIT = 14,
FID_CRYPTO_CIPHER_INIT_WITH_DATA = 15,
FID_CRYPTO_CIPHER_UPDATE = 16,
FID_CRYPTO_CIPHER_DOFINAL = 17,
FID_CRYPTO_SESSION_ABORT = 18,
FID_CRYPTO_GENERATE_KEY_PAIR = 19,
FID_CRYPTO_GENERATE_KEY_PAIR_BUFFER = 20,
FID_SECURITY_WRAP_OBJECT = 21,
FID_SECURITY_UNWRAP_OBJECT = 22,
FID_SECURITY_GET_SUID = 23,
FID_SECURITY_IS_DEVICE_BOUND = 24,
FID_SECURITY_BIND_DEVICE = 25,
FID_SYSTEM_GET_VERSION = 26,
FID_SECURITY_GET_TIME_STAMP = 27 }

Each function must have its own ID.

Functions

- _TLAPI_EXTERN_C tlApiResult_t callCryptoDriver (const marshalingParam_ptr pMarParam)

  Forwards call to Trustlet.

5.39.1 Define Documentation

5.39.1.1 #define MAX_MAR_LIST_LENGTH 8

Maximum number of parameter .

Maximum list of possible marshaling parameters.

5.39.2 Typedef Documentation

5.39.2.1 typedef void* void_ptr

a pointer to anything.

5.39.3 Enumeration Type Documentation

5.39.3.1 enum kpdFuncID_t

Each function must have its own ID.

Extend this list if you add a new function.

Enumerator:

FID_KPD_GRAB_KEYPAD Function to reserve the keypad.

FID_KPD_GET_C Function to read a character from the keypad.

FID_KPD_GET_MULTIPLE_C Function to read multiple characters from the keypad.

FID_KPD_RELEASE_KEYPAD Function to release the grabbed keypad.

5.39.3.2 enum cryptoFuncID_t

Each function must have its own ID.

Extend this list if you add a new function.

Enumerator:

FID_CRYPTO_MD ID for message digest algorithms.

FID_CRYPTO_MD_INIT Function to init a message digest.

FID_CRYPTO_MD_INIT_WITH_DATA Function to init a message digest with data.

FID_CRYPTO_MD_UPDATE Function to update a message digest.

FID_CRYPTO_MD_DOFINAL Function to finalize a message digest.

FID_CRYPTO_SIG ID for signature algorithms.

FID_CRYPTO_SIG_INIT Function to init a signature.

FID_CRYPTO_SIG_INIT_WITH_DATA Function to init a signature with data.

FID_CRYPTO_SIG_UPDATE Function to update a signature.

FID_CRYPTO_SIG_SIGN Function to make a signature.

FID_CRYPTO_SIG_VERIFY Function to verify a signature.

FID_CRYPTO_RNG ID for RNG algorithms.

FID_CRYPTO_RNG_GENERATE_DATA Function to generate random data.

FID_CRYPTO_CIPHER ID for cipher algorithms.

FID_CRYPTO_CIPHER_INIT Function to init a cipher.

FID_CRYPTO_CIPHER_INIT_WITH_DATA Function to init a cipher with data.

FID_CRYPTO_CIPHER_UPDATE Function to update a cipher.

FID_CRYPTO_CIPHER_DOFINAL Function to finalize a cipher.

FID_CRYPTO_SESSION_ABORT Function to abort a crypto session.

FID_CRYPTO_GENERATE_KEY_PAIR Function to generate a key pair.

FID_CRYPTO_GENERATE_KEY_PAIR_BUFFER Function to generate a key pair into a buffer.

FID_SECURITY_WRAP_OBJECT Function to wrap given data and create a secure object.

FID_SECURITY_UNWRAP_OBJECT Function to unwrap given secure object and create plaintext data.

FID_SECURITY_GET_SUID Function to acquire System on chip Unique ID.

FID_SECURITY_IS_DEVICE_BOUND Unused.

FID_SECURITY_BIND_DEVICE Unused.

FID_SYSTEM_GET_VERSION Function to get information about the underlying MobiCore version.

FID_SECURITY_GET_TIME_STAMP Function to get a secure time stamp.

5.39.4 Function Documentation

5.39.4.1 _TLAPI_EXTERN_C tlApiResult_t callCryptoDriver (const marshalingParam_ptr *pMarParam*)

Forwards call to Trustlet.

.Copies parameters from stack to shared memory .Calls Trustlet and returns response

Parameters

pMarParam	pointer to parameters marshalled in a struct	1whitetableShadegray

Returns

response of Crypto Driver to this operation.

Chapter 6

Data Structure Documentation

6.1 marshalingParam_t Struct Reference

Marshaled union.

`#include <TlApiMarshal.h>`

Data Fields

- uint32_t functionId

  Function identifier.

6.1.1 Detailed Description

Marshaled union.

6.1.2 Field Documentation

6.1.2.1 uint32_t marshalingParam_t::functionId

Function identifier.

The documentation for this struct was generated from the following file:

- TlApiMarshal.h

6.2 TestUtils::BypassDaemon::notification_t Struct Reference

Notification data structure.

Data Fields

- uint32_t sessionId

  Session ID.

- int32_t payload

  Additional notification information.

6.2.1 Detailed Description

Notification data structure.

6.2.2 Field Documentation

6.2.2.1 uint32_t TestUtils::BypassDaemon::notification_t::sessionId

Session ID.

6.2.2.2 int32_t TestUtils::BypassDaemon::notification_t::payload

Additional notification information.

The documentation for this struct was generated from the following file:

- TestUtils.cpp

6.3 tlApiCrAbort_t Struct Reference

Marshaled function parameters.

`#include <TlApiMarshal.h>`

6.3.1 Detailed Description

Marshaled function parameters. structs and union of marshaling parameters via TlApi.

Note

The structs can NEVER be packed !
The structs can NOT used via sizeof(..) !

The documentation for this struct was generated from the following file:

- TlApiMarshal.h

Chapter 7

File Documentation

7.1 MobiCoreDriverCmd.h File Reference

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1.

```
#include <inttypes.h>
#include "mcUuid.h"
#include "mcVersionInfo.h"
```

Defines

- #define MC_DEVICE_ID_DEFAULT 0

  The default device ID.

7.1.1 Detailed Description

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. 3. The name of the author may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR “AS IS” AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAM-

AGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

7.2 TlApiCrypto.c File Reference

TlApi security functions.

```
#include <stdint.h>
#include <stdio.h>
#include "TlApiMarshal.h"
#include "TlApi/TlApiCrypto.h"
#include "mcDriverId.h"
```

Enumerations

- enum bool

  Marshalling of Trustlet API crypto Function calls are mapped to a flat call.

7.2.1 Detailed Description

TlApi security functions.

7.3 TlApiMarshal.h File Reference

Marshaling types and declarations.

```
#include <stdint.h>
#include "TlApi/TlApiError.h"
#include "TlApi/TlApiCrypto.h"
#include "TlApi/TlApiKeypad.h"
#include "TlApi/TlApiSecurity.h"
#include "TlApi/TlApiMcSystem.h"
```

Data Structures

- struct tlApiCrAbort_t

Marshaled function parameters.

- struct marshalingParam_t

  Marshaled union.

Defines

- #define MAX_MAR_LIST_LENGTH 8

  Maximum number of parameter .

Typedefs

- typedef void * void_ptr

  a pointer to anything.

- typedef void_ptr addr_t

  an address, can be physical or virtual

- typedef addr_t buffer_t

  a data buffer

Enumerations

- enum kpdFuncID_t {
  FID_KPD_GRAB_KEYPAD = 0,
  FID_KPD_GET_C,
  FID_KPD_GET_MULTIPLE_C,
  FID_KPD_RELEASE_KEYPAD }

  Each function must have its own ID.

- enum cryptoFuncID_t {
  FID_CRYPTO_MD = 0,
  FID_CRYPTO_MD_INIT = 1,
  FID_CRYPTO_MD_INIT_WITH_DATA = 2,
  FID_CRYPTO_MD_UPDATE = 3,
  FID_CRYPTO_MD_DOFINAL = 4,
  FID_CRYPTO_SIG = 5,
  FID_CRYPTO_SIG_INIT = 6,
  FID_CRYPTO_SIG_INIT_WITH_DATA = 7,

FID_CRYPTO_SIG_UPDATE = 8,
FID_CRYPTO_SIG_SIGN = 9,
FID_CRYPTO_SIG_VERIFY = 10,
FID_CRYPTO_RNG = 11,
FID_CRYPTO_RNG_GENERATE_DATA = 12,
FID_CRYPTO_CIPHER = 13,
FID_CRYPTO_CIPHER_INIT = 14,
FID_CRYPTO_CIPHER_INIT_WITH_DATA = 15,
FID_CRYPTO_CIPHER_UPDATE = 16,
FID_CRYPTO_CIPHER_DOFINAL = 17,
FID_CRYPTO_SESSION_ABORT = 18,
FID_CRYPTO_GENERATE_KEY_PAIR = 19,
FID_CRYPTO_GENERATE_KEY_PAIR_BUFFER = 20,
FID_SECURITY_WRAP_OBJECT = 21,
FID_SECURITY_UNWRAP_OBJECT = 22,
FID_SECURITY_GET_SUID = 23,
FID_SECURITY_IS_DEVICE_BOUND = 24,
FID_SECURITY_BIND_DEVICE = 25,
FID_SYSTEM_GET_VERSION = 26,
FID_SECURITY_GET_TIME_STAMP = 27 }

Each function must have its own ID.

Functions

- _TLAPI_EXTERN_C tlApiResult_t callCryptoDriver (const marshalingParam_ptr pMarParam)

  Forwards call to Trustlet.

7.3.1 Detailed Description

Marshaling types and declarations. Functions for the marshaling of function ID and parameters.